N° D'ORDRE

THESE

présentée à

L'UNIVERSITE SCIENTIFIQUE ET MEDICALE DE GRENOBLE

pour obtenir

LE GRADE DE DOCTEUR DE TROISIEME CYCLE

"Informatique"

par

Jean-Pierre PEYRIN

ORGANISATION D'UN ESSAI DIDACTIQUE EN PROGRAMMATION ELEMENTAIRE

Thèse soutenue le 28 Janvier 1972 devant la commission d'examen

Monsieur	J. KUNTZMANN	Président
Messieurs	J.C. BOUSSARD	
	J. PERRIAULT	Examinateurs
	G. VEILLON	

UNIV.
LEIDEN
BIBL.

UNIVERSITE SCIENTIFIQUE
ET MEDICALE DE GRENOBLE

LISTE DES PROFESSEURS

Président : Monsieur Michel SOUTIF
Vice-Président : Monsieur Gabriel CAU

PROFESSEURS TITULAIRES

MM.	ANGLES D'AURIAC Paul	Mécanique des fluides
	ARNAUD Georges	Clinique des maladies infectieuses
	ARNAUD Paul	Chimie
	AYANT Yves	Physique approfondie
	BARBIER Jean-Claude	Physique expérimentale
	BARBIER Reynold	Géologie appliquée
	BARJON Robert	Physique nucléaire
	BARNOUD Fernand	Biosynthèse de la cellulose
	BARRIE Joseph	Clinique chirurgicale
	BENOIT Jean	Radioélectricité
	BESSON Jean	Electrochimie
	BEZES Henri	Chirurgie générale
	BLAMBERT Maurice	Mathématiques pures
	BOLLIET Louis	Informatique (IUT B)
	BONNET Georges	Electrotechnique
	BONNET Jean-Louis	Clinique ophtalmologique
	BONNET-EYMARD Joseph	Pathologie médicale
	BONNIER Etienne	Electrochimie Electrométallurgie
	BOUCHERLE André	Chimie et Toxicologie
	BOUCHEZ Robert	Physique nucléaire
	BRAVARD Yves	Géographie
	BRISSONNEAU Pierre	Physique du Solide
	BUYLE-BODIN Maurice	Electronique
	CABANAC Jean	Pathologie chirurgicale
	CABANEL Guy	Clinique rhumatologique et hydrologique
	CALAS François	Anatomie
	CARRAZ Gilbert	Biologie animale et pharmacodynamie
	CAU Gabriel	Médecine légale et Toxicologie
	CAUQUIS Georges	Chimie organique
	CHABAUTY Claude	Mathématiques pures
	CHATEAU Robert	Thérapeutique
	CHENE Marcel	Chimie papetière
	COEUR André	Pharmacie chimique
	CONTAMIN Robert	Clinique gynécologique
	COUDERC Pierre	Anatomie Pathologique
	CRAYA Antoine	Mécanique
Mme	DEBELMAS Anne-Marie	Matière médicale
MM.	DEBELMAS Jacques	Géologie générale
	DEGRANGE Charles	Zoologie
	DESSAUX Georges	Physiologie animale
	DODU Jacques	Mécanique appliquée
	DREYFUS Bernard	Thermodynamique
	DUCROS Pierre	Cristallographie
	DUGOIS Pierre	Clinique de Dermatologie et Syphiligraphie

	FAU René	Clinique neuro-psychiatrique
	FELICI Noël	Electrostatique
	GAGNAIRE Didier	Chimie physique
	GALLISSOT François	Mathématiques pures
	GALVANI Octave	Mathématiques pures
	GASTINEL Noël	Analyse numérique
	GERBER Robert	Mathématiques pures
	GIRAUD Pierre	Géologie
	KLEIN Joseph	Mathématiques pures
Mme	KOFLER Lucie	Botanique et physiologie végétale
MM.	KOSZUL Jean-Louis	Mathématiques pures
	KRAVTCHENKO Julien	Mécanique
	KUNTZMANN Jean	Mathématiques appliquées
	LACAZE Albert	Thermodynamique
	LACHARME Jean	Biologie végétale
	LATURAZE Jean	Biochimie pharmaceutique
	LEDRU Jean	Clinique médicale B
	LLIBOUTRY Louis	Géophysique
	LOUP Jean	Géographie
Mle	LUTZ Elisabeth	Mathématiques pures
MM.	MALGRANGE Bernard	Mathématiques pures
	MALINAS Yves	Clinique obstétricale
	MARTIN-NOEL Pierre	Séméiologie médicale
	MASSEPORT Jean	Géographie
	MAZARE Yves	Clinique médicale A
	MICHEL Robert	Minéralogie et Pétrographie
	MOURIQUAND Claude	Histologie
	MOUSSA André	Chimie nucléaire
	NEEL Louis	Physique du Solide
	OZENDA Paul	Botanique
	PAUTHENET René	Electrotechnique
	PAYAN Jean-Jacques	Mathématiques pures
	PEBAY-PEYROULA Jean-Claude	Physique
	PERRET René	Servomécanismes
	PILLET Emile	Physique industrielle
	ROUSSAT André	Chimie systématique
	RENARD Michel	Thermodynamique
	REULOS René	Physique industrielle
	RINALDI Renaud	Physique
	ROGET Jean	Clinique de pédiatrie et de puériculture
	SANTON Lucien	Mécanique
	SEIGNEURIN Raymond	Microbiologie et Hygiène
	SENGEL Philippe	Zoologie
	SILBERT Robert	Mécanique des fluides
	SOUTIF Michel	Physique générale
	TANCHE Maurice	Physiologie
	TRAYNARD Philippe	Chimie générale
	VAILLAND François	Zoologie
	VAUQUOIS Bernard	Calcul électronique
Mme	VERAIN Alice	Pharmacie galénique
	VERAIN André	Physique
Mme	VEYRET Germaine	Géographie
MM.	VEYRET Paul	Géographie
	VIGNAIS Pierre	Biochimie médicale
	YOCCOZ Jean	Physique nucléaire théorique

PROFESSEURS ASSOCIES

MM.	BULLEMER Bernhard	Physique
	RADHAKRISHNA Pidatala	Thermodynamique

PROFESSEURS SANS CHAIRE

MM.	AUBERT Guy	Physique
Mme	BARBIER Marie-Jeanne	Electrochimie
MM.	BARRA Jean	Mathématiques appliquées
	BEAUDOING André	Pédiatrie
	BERTRANDIAS Jean-Paul	Mathématiques appliquées
	BIAREZ Jean-Pierre	Mécanique
	BONNETAIN Lucien	Chimie minérale
Mme	BONNIER Jane	Chimie générale
MM.	CARLIER Georges	Biologie végétale
	COHEN Joseph	Electrotechnique
	COUMES André	Radioélectricité
	DEPASSEL Roger	Mécanique des Fluides
	DEPORTES Charles	Chimie minérale
	DESRE Pierre	Métallurgie
	DOLIQUE Jean-Michel	Physique des plasmas
	GAUTHIER Yves	Sciences biologiques
	GEINDRE Michel	Electroradiologie
	GIDON Paul	Géologie et Minéralogie
	GLENAT René	Chimie organique
	HACQUES Gérard	Calcul numérique
	JANIN Bernard	Géographie
Mme	KAHANE Josette	Physique
MM.	LATREILLE René	Chirurgie générale
	LAURENT Pierre	Mathématiques appliquées
	MULLER Jean-Michel	Thérapeutique
	PERRIAUX Jean-Jacques	Géologie et minéralogie
	POULOUJADOFF Michel	Electrotechnique
	REBECQ Jacques	Biologie (CUS)
	REVOL Michel	Urologie
	REYMOND Jean-Charles	Chirurgie générale
	ROBERT André	Chimie papetière
	SARRAZIN Roger	Anatomie et chirurgie
	SARROT-REYNAULD Jean	Géologie
	SIBILLE Robert	Construction Mécanique
	SIROT Louis	Chirurgie générale
Mme	SOUTIF Jeanne	Physique générale
M.	VALENTIN Jacques	Physique nucléaire

MAITRES DE CONFERENCES ET MAITRES DE CONFERENCES AGREGES

Mle	AGNIUS-DELORD Claudine	Physique pharmaceutique
	ALARY Josette	Chimie analytique
MM.	AMBLARD Pierre	Dermatologie
	AMBROISE-THOMAS Pierre	Parasitologie
	ARMAND Yves	Chimie

	BEGUIN Claude	Chimie organique
	BELORIZKY Elie	Physique
	BENZAKEN Claude	Mathématiques appliquées
Mme	BERTRANDIAS Françoise	Mathématiques pures
MM.	BLIMAN Samuel	Electronique (EIE)
	BLOCH Daniel	Electrotechnique
Mme	BOUCHE Liane	Mathématiques (CUS)
MM.	BOUCHET Yves	Anatomie
	BOUSSARD Jean-Claude	Mathématiques appliquées
	BOUVARD Maurice	Mécanique des Fluides
	BRIERE Georges	Physique expérimentale
	BRODEAU François	Mathématiques (IUT B)
	BRUGEL Lucien	Energétique
	BUISSON Roger	Physique
	BUTEL Jean	Orthopédie
	CHAMBAZ Edmond	Biochimie médicale
	CHAMPETIER Jean	Anatomie et organogénèse
	CHARACHON Robert	Oto-Rhino-Laryngologie
	CHIAVERINA Jean	Biologie appliquée (EFP)
	CHIBON Pierre	Biologie animale
	COHEN-ADDAD Jean-Pierre	Spectrométrie physique
	COLOMB Maurice	Biochimie médicale
	CONTE René	Physique
	CROUZET Guy	Radiologie
	DURAND Francis	Métallurgie
	DUSSAUD René	Mathématiques (CUS)
Mme	ETERRADOSSI Jacqueline	Physiologie
MM.	FAURE Jacques	Médecine légale
	GAVEND Michel	Pharmacologie
	GENSAC Pierre	Botanique
	GERMAIN Jean-Pierre	Mécanique
	GIDON Maurice	Géologie
	GRIFFITHS Michael	Mathématiques appliquées
	GROULADE Joseph	Biochimie médicale
	HOLLARD Daniel	Hématologie
	HUGONOT Robert	Hygiène et médecine préventive
	IDELMAN Simon	Physiologie animale
	IVANES Marcel	Electricité
	JALBERT Pierre	Histologie
	JOLY Jean-René	Mathématiques pures
	JOUBERT Jean-Claude	Physique du Solide
	JULLIEN Pierre	Mathématiques pures
	KAHANE André	Physique générale
	KUHN Gérard	Physique
Mme	LAJZEROWICZ Jeannine	Physique
MM.	LAJZEROWICZ Joseph	Physique
	LANCIA Roland	Physique atomique
	LE JUNTER Noël	Electronique
	LEROY Philippe	Mathématiques
	LOISEAUX Jean-Marie	Physique nucléaire
	LONGEQUEUE Jean-Pierre	Physique nucléaire
	LUU DUC Cuong	Chimie organique
	MACHE Régis	Physiologie végétale
	MAGNIN Robert	Hygiène et Médecine préventive
	MARECHAL Jean	Mécanique
	MARTIN-BOUYER Michel	Chimie (CUS)
	MAYNARD Roger	Physique du Solide
	MICOUD Max	Maladies infectieuses
	MOREAU René	Hydraulique (INP)

	NEGRE Robert	Mécanique
	PARAMELLE Bernard	Pneumologie
	PECCOUD François	Analyse (IUT B)
	PEFFEN René	Métallurgie
	PELMONT Jean	Physiologie animale
	PERRET Jean	Neurologie
	PERRIN Louis	Pathologie expérimentale
	PFISTER Jean-Claude	Physique du Solide
	PHELIP Xavier	Rhumatologie
Mle	PIERY Yvette	Biologie animale
	RACHAIL Michel	Médecine interne
	RACINET Claude	Gynécologie et obstétrique
	RICHARD Lucien	Botanique
Mme	RINAUDO Marguerite	Chimie macromoléculaire
MM.	ROMIER Guy	Mathématiques (IUT B)
	ROUGEMONT (DE) Jacques	Neuro-chirurgie
	STIEGLITZ Paul	Anesthésiologie
	STOEBNER Pierre	Anatomie pathologique
	VAN CUTSEM Bernard	Mathématiques appliquées
	VEILLON Gérard	Mathématiques appliquées (INP)
	VIALON Pierre	Géologie
	VOOG Robert	Médecine interne
	VROUSSOS Constantin	Radiologie
	ZADWORNY François	Electronique

MAITRES DE CONFERENCES ASSOCIES

MM.	BOUDOURIS Georges	Radioélectricité
	CHEEKE John	Thermodynamique
	GOLDSCHMIDT Hubert	Mathématiques
	YACOUD Mahmoud	Médecine légale

CHARGES DE FONCTIONS DE MAITRES DE CONFERENCES

Mme	BERIEL Hélène	Physiologie
Mme	RENAUDET Jacqueline	Microbiologie

Je tiens à remercier

Monsieur le professeur Jean KUNTZMANN, directeur de l'Institut de Mathématiques Appliquées de Grenoble qui m'a fait l'honneur de présider le Jury de Thèse, et dont les encouragements et les conseils m'ont été précieux.

Monsieur le professeur Jacques PERRIAULT, directeur du département de recherche et d'information à l'Office Français des Techniques Modernes d'Education, qui a bien voulu accorder une réelle attention à mon travail et accepter de faire partie du Jury.

Monsieur Jean-Claude BOUSSARD, professeur à l'Université Scientifique et Médicale de Grenoble, qui a orienté et encouragé mes recherches et m'a toujours témoigné sa confiance.

Monsieur Gérard VEILLON, maître de conférences à l'Institut National Polytechnique de Grenoble, qui s'est intéressé, avec sympathie, à mon travail.

Je veux également exprimer toute ma reconnaissance à Jean-Pierre PAILLARD et Jacques VOIRON, assistants à l'Institut Universitaire de Technologie de Grenoble, pour leur aide amicale.

Enfin, je remercie Madame Renée CARRE-PIERRAT, Messieurs Claude ANGUILLE, Claude LABORIE, et Paul MOUNET, auxquels je dois la réalisation matérielle de cet ouvrage.

Digitized by the Internet Archive
in 2026

https://archive.org/details/IA41534904_0025

Or, ce qu'il y a de nouveau dans l'histoire, c'est que la connaissance peut être davantage qu'un moyen.

NIETZSCHE (Le gai savoir).

INTRODUCTION

A partir d'un enseignement reçu, puis donné, il m'a semblé intéressant d'étudier les problèmes de l'enseignement de la programmation élémentaire.

Le premier travail a été d'analyser ce que j'appellerai " l'enseignement existant " pour en connaître précisément les composantes et pour en déterminer les effets quant à la suite donnée à cet enseignement d'une part, quant à la connaissance acquise par un futur informaticien d'autre part.

Cette analyse critique étant faite, le second travail a été la définition d'un cours possible, c'est à dire, essentiellement, l'énumération des connaissances fondamentales à enseigner dans un premier temps.

L'essentiel était fait, au niveau théorique ;

Il fallait enseigner.

Les conclusions émises ne valent que par les expérimentations effectuées. Et plusieurs expérimentations ont été effectuées - Semblables quant à l'esprit, mais souvent différentes quant à la forme - Cette non-identité dans des démarches parallèles est une source de critiques profitables à l'enseignement.

Le projet OEDIPE est né vers le mois de septembre 1969 à partir des premières conclusions. OEDIPE propose l'Organisation d'un Essai Didactique en Programmation Elémentaire. Ce projet s'intéresse non seulement à la détermination des notions élémentaires - composantes de la programmation dite élémentaire - mais aussi à la détermination des moyens d'enseignements - utilisation des moyens existants et création de moyens nouveaux - dans une nouvelle optique de la relation maître-élève.

La première réalisation de ce projet est le système SCALP - appelé ainsi parce que ce système est fondé sur une Simulation d'un Calculateur Pédagogique (l'adjectif pédagogique s'appliquant aussi bien au mot calculateur qu'au mot simulation).

Le texte, que je vous propose de lire maintenant, a pour objet essentiel la description du système SCALP - (chapitre 2).

La description de ce système est précédée d'une brève étude des problèmes de l'enseignement de la programmation élémentaire dont la " conclusion " est une description succinte du projet OEDIPE - (chapitre 1).

La description du système SCALP est conclue par une description de l'expérimentation effectuée à la faculté des Sciences de Grenoble - (chapitre 3).

Je ne suis pas le seul à porter un jugement critique sur l'enseignement existant.- Les ouvrages que j'ai pu lire [B - 8. 20. 21. 22. 26. 35] qui, implicitement ou explicitement, tentaient une réforme, m'ont prouvé que je n'étais pas non plus le seul à proposer un remède à ses imperfections.

Mais ces travaux ne présentent pas, avec mon système, de lien suffisamment précis ou étroit pour que je puisse établir un parallèle en cours de description.

C'est pourquoi j'ai estimé plus juste de dresser une table de références générales.

TABLE DES MATIERES

CHAPITRE I - L'ENSEIGNEMENT DE LA PROGRAMMATION ELEMENTAIRE - 1

I § 1 - Généralités sur l'enseignement de la programmation élémentaire. 3
- sur le fond de l'enseignement
- sur la forme de l'enseignement.

I § 2 - Regard sur l'enseignement existant 6

I § 3 - Une nouvelle approche de l'enseignement de la programmation élémentaire. 10
- introduction
- un enseignement de concepts et de méthodes.
- le projet OEDIPE.

CHAPITRE II - LE SYSTEME SCALP. 21

II § 1 - Présentation du système SCALP 22
- finalité du cours
- durée du cours.
- les "outils" utilisés.
- la population enseignée.

II § 2 - Description du système SCALP. 25
- analyse diachronique. 26
- analyse synchronique. 38
  - temps de sensibilisation. 39
  - temps d'adaptation. 43
  - temps d'acquisition. 45
  - temps de synthèse. 59

II § 3 - Schémas synoptiques. 63

II § 4 - Evaluation du coût de l'implantation du système. 66

II § 5 - Subséquence. 67
- un contrôle automatique des connaissances.
- la disponibilité de l'ordinateur.

CHAPITRE III - EXPERIMENTATION - 71

III § 1 - L'ordinateur 72

III § 2 - Le support software. 80

III § 3 - Organigramme logique. 82

III § 4 - Les phases. 95
- la phase NUMER.
- la phase EXER.
- la phase PROCES.
- la phase MONIT.

III § 5 - Bilan 110
- deux années d'expérimentations.
- une expérimentation définitive.

CHAPITRE I

L'ENSEIGNEMENT DE LA PROGRAMMATION ELEMENTAIRE

-- Généralités

-- Regard sur l'enseignement existant

-- Une nouvelle approche de l'enseignement

-- Le projet OEDIPE

L'enseignement de la programmation n'est qu'un cas particulier de l'enseignement. Il est concerné, comme tout autre, par la rénovation pédagogique.

Ce chapître est volontairement bref : le texte " Regard sur l'enseignement - Motivation pédagogique du projet OEDIPE " A3 décrit l'évolution de l'enseignement, des méthodes " traditionnelles " à la rénovation pédagogique, à travers les mutations de la société contemporaine.

Ce qui y est dit sur l'enseignement, en général, est vrai pour l'enseignement de la programmation, en particulier.

Nous ne reviendrons pas sur ce qu'est la mutation dans le monde enseignant par rapport à la mutation en dehors de ce monde enseignant.

Nous ne reviendrons pas sur les principes établis par la rénovation pédagogique, quant à la définition de la connaissance, quant à la relation outils-élèves.

Mon propos, dans ce chapître est d'exposer les faits tels que j'ai pu les observer ou les ressentir dans ma vie d'étudiant puis d'enseignant.

Ils n'ont, peut-être, aucune valeur scientifique, mais ils ont, sûrement, valeur de témoignage. Ils ne sont pas " objet " de ma thèse, mais sa motivation pédagogique profonde ; c'est pourquoi je tiens à les décrire.

I § 1 - GENERALITES SUR L'ENSEIGNEMENT DE LA PROGRAMMATION ELEMENTAIRE -

- Sur le fond de l'enseignement :

La programmation est une matière nouvelle, mais cette matière se rattache à certaines connaissances :

- soit acquises : - processus de démonstration (relation de cause à effet).
  - automate.
- soit intuitives : - fonction de mémoire.
  - langage (agent de communication).

Son enseignement doit susciter chez l'élève la découverte des analogies entre ses connaissances antérieures et les nouveaux concepts.

Son enseignement doit, alors, permettre la préhension des singularités.

L'élève a besoin d'être dirigé dans la découverte de l'objet. Etant donné ses connaissances préalables au cours, sa découverte (d) est analogue à " la " découverte (D) de l'objet.

D
- Au départ, il y a la motivation.
- Cette motivation implique la création.
- L'objet étant " créé ", il faut analyser les modes d'utilisation.
- L'utilisation implique l'évolution.

La programmation est essentiellement manipulatoire.

Son enseignement doit permettre à l'élève, à chaque instant, de maîtriser les nouvelles connaissances.

d
- L'acquisition d'un concept vi pratique sommaire.
- Cette acquisition permet un des concepts.
- La rétention se forme à par cette pratique.
- L'évolution se base sur la ention.

La connaissance préalable - l'avant-cours - comme la connaissance finale - l'après-cours - sont des connaissances objectives.

La progression de l'avant-cours à l'après-cours est, à priori, établie sur une suite de connaissances subjectives.

Permettre la maîtrise permanente des faits particuliers, c'est faire de chaque partie du cours, un nouvel " avant-cours " pour la suite, c'est donc instituer une progression établie sur une suite de connaissances objectives.

cf Figure I 1

- Sur la forme de l'enseignement :

Les concepts primitifs ne sont pas indépendants.

L'élève doit acquérir une connaissance globale de la programmation pour comprendre les faits particuliers, et il ne peut comprendre réellement la programmation s'il n'a pas maîtrisé chaque point particulier.

Sur le plan pédagogique, c'est :
La dialectique des méthodes globale et analytique.

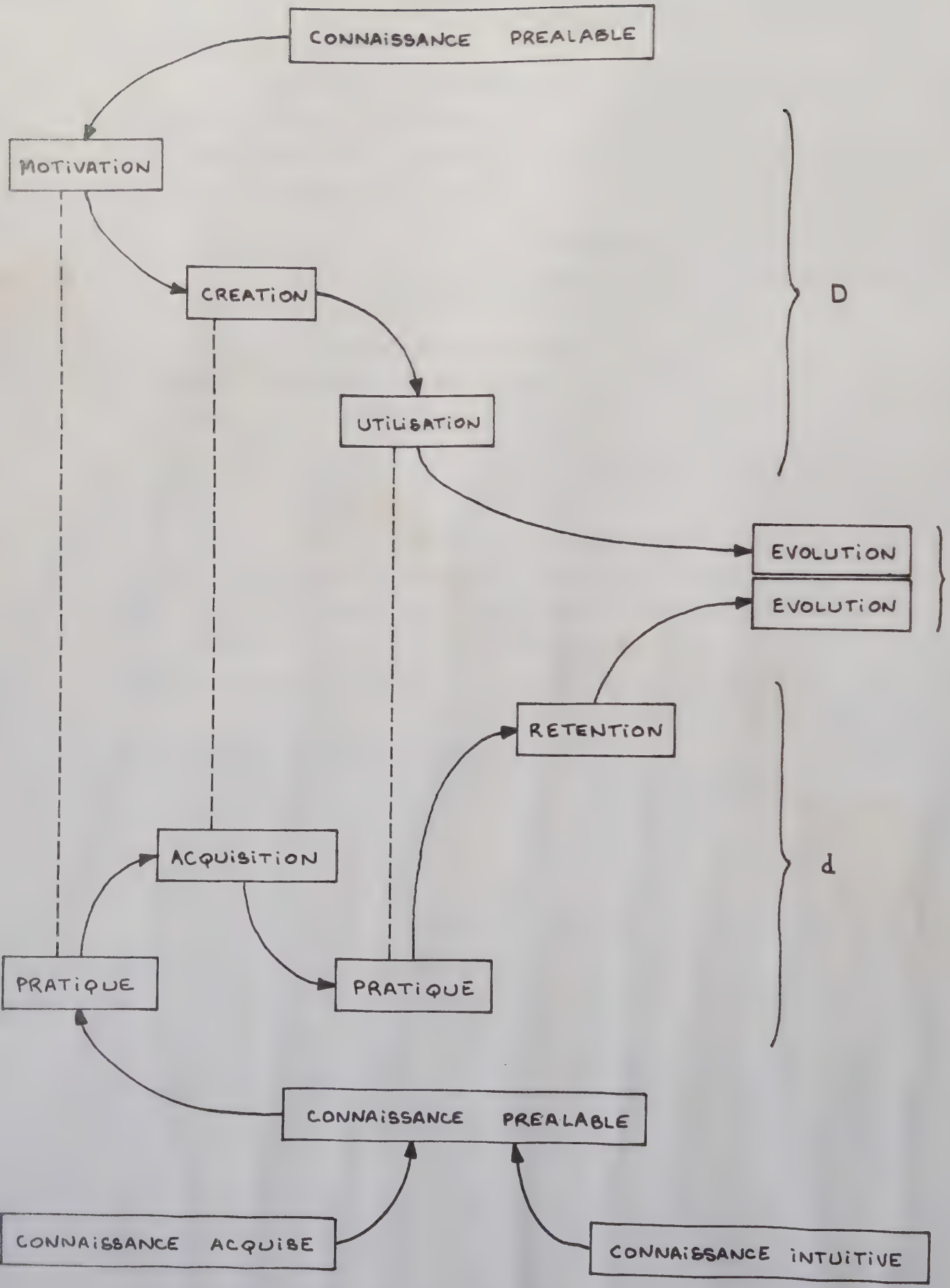
Un corollaire est que, les concepts primitifs, donc ni essentiellement descriptifs, ni essentiellement déductifs, ne peuvent être mis en pratique sur un ordinateur, quelqu'il soit, tant que celui-ci n'a pas été compris globalement.

Mais cette acquisition globale ne peut être que pratique.

Deux remarques sont alors importantes :

- Il est toujours possible de décomposer un ordinateur en " composantes ".
  Une première acquisition globale, c'est l'acquisition de l'ordinateur en tant que somme des composantes.
  Chaque composante se subdivise. Et son acquisition globale regroupe les " sous-composantes ".
  Et ainsi de suite, jusqu'aux éléments fondamentaux de l'ordinateur.
- L'acquisition analytique d'un " petit " ordinateur débouche, naturellement, sur l'acquisition globale d'un ordinateur plus complexe.

FIGURE I.1.


CONNAISSANCE PREALABLE
MOTIVATION
CREATION
UTILISATION
D
EVOLUTION
EVOLUTION
RETENTION
ACQUISITION
d
PRATIQUE
PRATIQUE
CONNAISSANCE PREALABLE
CONNAISSANCE ACQUISE
CONNAISSANCE INTUITIVE

I § 2 - REGARD SUR L'ENSEIGNEMENT EXISTANT -

L'enseignement de la programmation - matière nouvelle - a pris, à son origine, et a conservé, les structures de l'enseignement " traditionnel ".

D'autre part, le propos de cet enseignement est essentiellement l'utilisation de l'ordinateur - l'enseignement est essentiellement un enseignement de langages de programmation - oubliant que l'utilisation de l'ordinateur est une conséquence de la nature de cet ordinateur.

Généralement, le premier cours de programmation (où sont enseignées les premières notions de programmation) est donné sur un certain ordinateur.

En fait, il ne s'agit pas d'un cours sur cet ordinateur (donc sur " un " ordinateur), mais d'un cours sur le software conçu pour cet ordinateur.

Il faut noter - et c'est important - que ce software est utilisé aussi bien pour le premier programme du débutant, que pour le programme du chercheur manipulant les ordinateurs depuis déjà de nombreuses années.

Le cours est généralement divisé en deux parties :

. La première, très rapide, où l'on donne quelques généralités sur l'ordinateur (c'est un exposé " magistral " sur la représentation de l'information, le support d'information, le principe de l'adressage, les instructions, les registres, le traitement de l'information, et, enfin, l'utilisation du software).

Dans cette " description ", on " expose " deux concepts essentiels de la programmation :

- la structure d'un ordinateur.
- la structure d'un système.

. La deuxième est plus longue :
Les élèves élaborent quelques programmes.

Nous constatons les faits suivants :

. Lors des premières séances, l'élève ne raisonne pas. Il écrit son programme, empiriquement, dans le langage enseigné, sans comprendre la réalité pratique de ce qu'il écrit.

Il ne sait pas (ou mal) ce qu'est un algorithme et ne comprend que difficilement ce qu'est un organigramme (la plupart du temps, d'ailleurs, on lui fournit une " sorte " d'organigramme, et son travail se borne à un exercice de traduction.

Il apprend, peu à peu, à manier le langage, mais sans méthode précise.

Et s'il doit apprendre la manipulation d'un autre ordinateur, il retrouvera des problèmes du même ordre.

. L'élève, lors du passage de son programme en machine, a comme difficulté essentielle non pas tant la compréhension logique de son programme, mais la compréhension du software utilisé pour le passage (système d'exploitation - assemblage - chargement - exécution).

Il ne sait pas ce qu'est un assembleur ; il ne comprend ni sa nature, ni son rôle.

Il ne sait que très mal lire un " dump ".

Il ne sait pas (mais il ne le peut pas) intelligemment comprendre et corriger ses erreurs.

. Etant donné la complexité des systèmes utilisés (systèmes évidemment non pédagogiques), plusieurs notions fondamentales (telles les entrées-sorties, les interruptions...) ne sont pas abordées, alors que plusieurs notions moins primordiales sont traitées (notions propres aux instructions particulières de l'ordinateur, notions propres aux particularités - commodités - d'utilisation du système).

Donnons un exemple :

Généralement, le cours propose diverses macro-instructions pour permettre à l'élève d'effectuer, dans ses programmes, des opérations de transfert d'informations.

La macro-instruction est présentée comme une " recette " :

- tel symbole doit être écrit à tel endroit.
- tels paramètres doivent être indiqués de telle façon.
- Ces paramètres doivent être écrits à tels endroits.

L'opération élémentaire d'entrées-sorties est complètement masquée par l'usage d'une macro-instruction (la notion d'entrées-sorties est pourtant une notion fondamentale).

Le principe d'une macro-instruction (beaucoup moins fondamental que le principe des entrées-sorties)n'est même pas explicité. (S'il n'est pas fondamental, il est cependant intéressant).

Qu'apprennent donc les élèves ?

Certainement pas le fonctionnement, ne serait-ce que global, d'un ordinateur.

Les concepts de base, nous l'avons vu, sont " exposés " en un temps extrêmement bref et sans aucune application pratique immédiate.

Ils apprennent, essentiellement, les régles d'écriture d'un langage au service de quelques techniques de la programmation.

Ainsi, de syntaxe en syntaxe, les élèves comprennent, ou ne comprennent pas, la programmation.

I § 3 - UNE NOUVELLE APPROCHE DE L'ENSEIGNEMENT DE LA PROGRAMMATION ELEMENTAIRE -

1°) - INTRODUCTION -

. Un fait historique :

En dix ans, la population enseignée a plus que décuplé.

En dix ans, la conception des ordinateurs a considérablement évolué.

Corrélativement, le travail des programmeurs a dû s'élargir et empiriquement se structurer.

Les connaissances nécessaires aux futurs utilisateurs se sont accrues au contact de domaines d'emplois toujours nouveaux : ce qui demande à l'élève un travail plus intensif et encore assez disparate.

En somme, l'informatique était un " artisanat ", c'est devenu une " industrie ".

Si le contenu de l'enseignement s'est développé, il n'a, néanmoins, pas encore acquis d'unité profonde ; aussi les méthodes d'enseignement sont restées sensiblement les mêmes.

On donne toujours un enseignement de " recettes ".

Par exemple, l'élève sait utiliser un ou deux ordinateurs particuliers, mais ne comprend pas exactement ce qu'est " un ordinateur ".

Il faut, maintenant, un enseignement de " concepts " et de " méthodes ".

. Un fait pédagogique :

Un étudiant de l'université qui débute en informatique, est vierge en programmation, comme est vierge en algèbre un enfant de six ans entrant au cours préparatoire.

Et si, étant donné les différences d'âges et de cultures, l'enseignement ne peut se présenter de la même façon, à tout niveau, le principe de l'enseignement ne peut qu'être le même :

. analogies entre les connaissances antérieures et les nouveaux concepts.

. découverte progressive et concrète des singularités.

. synthèse des faits particuliers.

L'université doit maintenant s'intéresser activement à la pédagogie.

L'université doit considérer que, pour l'élève, le processus de compréhension des connaissances est finalement plus utile que la connaissance elle-même, puisque cette connaissance est " mobile ".

C'est à dire que les enseignants doivent se soucier plus de la manière dont les élèves reçoivent ce qu'ils leur donnent, que de ce qu'ils leur donnent.

2°) - UN ENSEIGNEMENT DE CONCEPTS ET DE METHODES -

Nous l'avons vu, l'évolution historique et corrélativement l'évolution pédagogique imposent de nouvelles structures à l'enseignement de la programmation.

L'université doit-elle former des techniciens rentables pendant une ou deux générations d'ordinateurs (une dizaine d'année) par un enseignement " rapide " des langages usuels et des techniques usuelles !

Est-il d'ailleurs possible, en informatique, à l'heure actuelle, de parler de langages usuels et de techniques usuelles ?

On parle beaucoup d'opérateurs, de programmeurs, d'analystes, d'experts,....

Tous sont informaticiens, d'une façon ou d'une autre.

L'université doit-elle, dès le tout début d'une formation, tenir compte de telles distinctions ?

Ce "tout-début " ne devrait-il pas être " commun " à tous les futurs informaticiens ?

Un point de départ, au moins, est commun :

- C'est l'ordinateur.

Non pas l'ordinateur actuel-outil actuel ; mais l'ordinateur en tant que concept : un outil traitant une quantité d'informations quelque soit la nature de ces informations, quelque soit la nature du support de ces informations, quelque soit le mode de traitement de ces informations.

L'enseignement de la programmation joue un grand rôle dans la formation de l'informaticien.

Cet enseignement doit être scindé en deux parties :

- Un enseignement élémentaire global, tout d'abord.

- Un enseignement de spécialisation, ensuite.

L'enseignement doit donner, à l'élève, les méthodes d'apprentissage plus que l'apprentissage.

L'enseignement doit lui donner les fondements théoriques et pratiques nécessaires à tout travail immédiat ainsi qu'à tout " recyclage " - inévitable - qu'il soit individuel ou universitaire.

Que doit être cet enseignement préliminaire ?

Contentons nous, pour l'instant, d'examiner ce que doit-être un enseignement devant précéder l'enseignement de l'utilisation " réelle " d'un ordinateur. (C'est celui que nous avons critiqué - cet enseignement d'un langage de programmation pour l'élaboration d'un certain nombre de problèmes).

Deux thèmes, au départ, sont fondamentaux :

- La structure d'un ordinateur et l'analyse d'un problème.

- La structure d'un ordinateur :

Connaître, donc comprendre, la structure d'un ordinateur, c'est réaliser, concrètement, d'une part ce qu'est l'information (support, circuits, codage, accès), d'autre part ce qu'est un mode de traitement de cette information (processeur, registres, circuits, décodage, exécution).

Connaissant la nature de l'information et le principe du traitement de cette information, l'élève maîtrise le langage machine (binaire).

La connaissance de ce langage machine lui permet de réaliser ce que peut être un langage de programmation - lui permet de réaliser l'existence et le travail d'un software.

- L'analyse d'un problème.

Savoir analyser un problème, c'est savoir découvrir un processus de résolution. C'est alors savoir le " rédiger " de façon explicite.

Sachant déterminer les processus de calcul, l'élève peut aborder les aspects fondamentaux de l'algorithmique.

Cf. Figure I 2.

Il est bien entendu que ces deux thèmes sont enseignés conjointement : ce ne sont pas deux enseignements indépendants.

Ces deux "enseignements" ont des finalités complémentaires, donc des exécutions corrélées.

Nous avons parlé de théorie et de pratique :

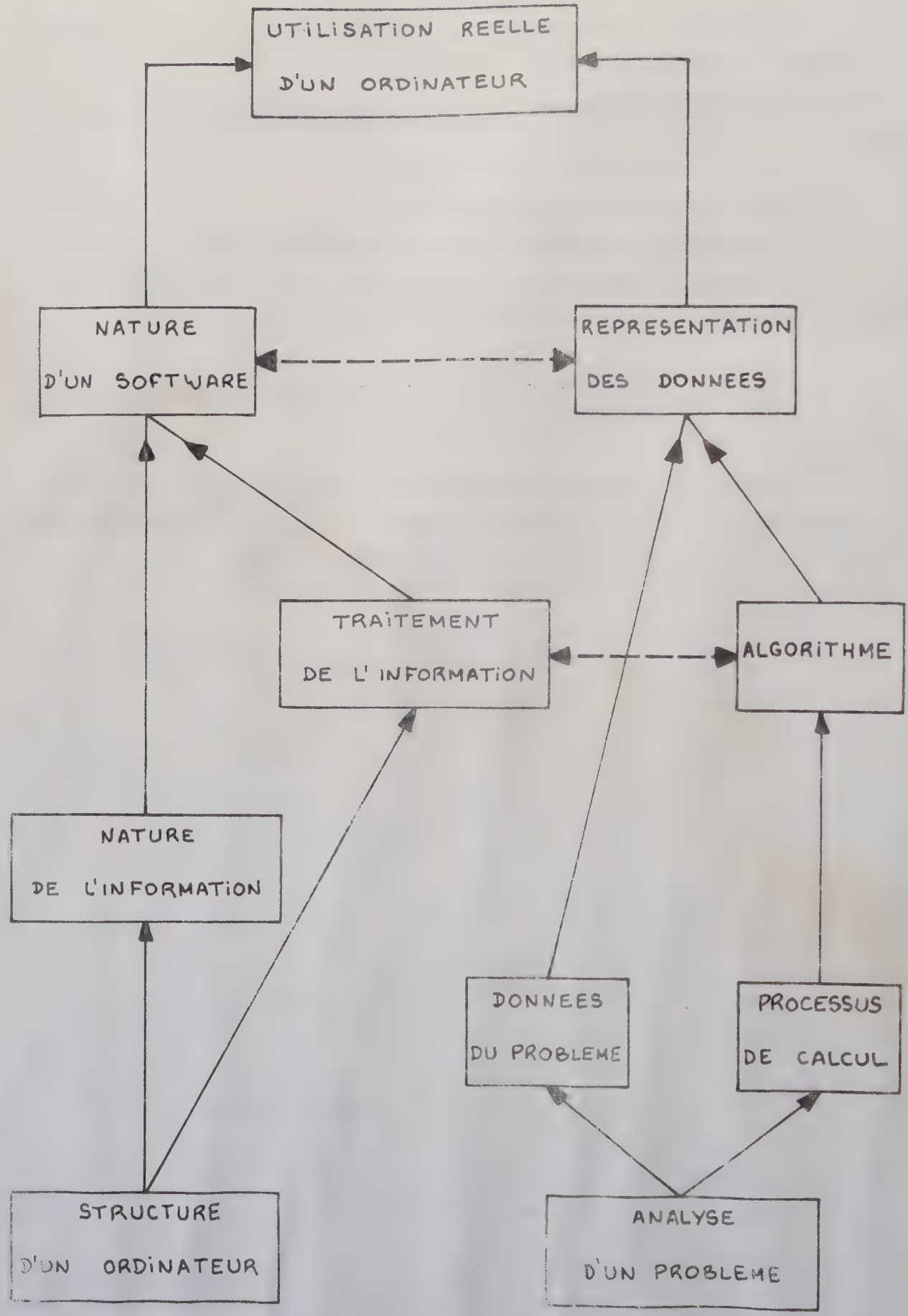
Cet enseignement désire donner un fondement théorique à l'élève.

Mais cette "théorie" ne s'acquiert que par une pratique régulière en tout domaine.

Nos deux thèmes ne peuvent donc être enseignés que par la manipulation pratique sur les ordinateurs actuels.

La synthèse permanente entre ces deux thèmes permet le dépassement de l'ordinateur actuel pour aboutir précisemment, au niveau théorique souhaité.

FIGURE I.2.


UTILISATION REELLE
D'UN ORDINATEUR
NATURE
D'UN SOFTWARE
REPRESENTATION
DES DONNEES
TRAITEMENT
DE L'INFORMATION
ALGORITHME
NATURE
DE L'INFORMATION
DONNEES
DU PROBLEME
PROCESSUS
DE CALCUL
STRUCTURE
D'UN ORDINATEUR
ANALYSE
D'UN PROBLEME

I § 4 - LE PROJET OEDIPE -

1°) - INTRODUCTION.

Organiser un enseignement annuel de la programmation élémentaire :

- C'est, tout d'abord élaborer une méthode :
  - étudier la finalité d'un tel enseignement.
  - rechercher puis analyser l'ensemble des notions à enseigner pour atteindre cette finalité.
  - définir une démarche pédagogique.

- C'est, ensuite, réaliser cette méthode :
  - effectuer un " système " d'enseignement.

- C'est, enfin, mettre en question le " système "
  - critiquer le système conçu en fonction d'une (ou plusieurs) expérimentation (s).
  - rénover le système en tenant compte de ces critiques.
  - etc....

2°) - L'ELABORATION D'UNE METHODE

a) Etude de la finalité :

Généralement, on dit :
" les élèves doivent savoir bien programmer " -
Mais que signifie " bien savoir programmer " ?
Et programmer quoi ? Et comment ?

- Des commissions, depuis quelques temps, se réunissent pour tenter de préciser les finalités d'une maîtrise d'informatique, d'une école d'ingénieur, d'un institut universitaire de technologie, d'un institut de programmation.

Et c'est le premier travail à accomplir.

Les recherches actuelles ont déjà permis une modification des anciennes structures et si l'on songe que l'essentiel de ces recherches, pour l'instant, s'est traduit par des paroles parfois fixées sur le papier, il n'est pas utopique de penser qu'il y aura de plus en plus de " papiers ", et pas trop utopique de penser que ces " papiers " conduiront à de nouvelles modifications profondes.

- Le projet OEDIPE ne s'intéresse pas à un enseignement complet de la programmation, mais à ce qui serait une première année des enseignements pré-cités. Cela ne veut pas dire qu'il ne tient pas compte de l'existence de ces enseignements complets ; cela ne veut pas dire qu'il ne tient pas compte des travaux des commissions.

- Le postulat du projet OEDIPE :

Quelque soit la spécialisation future d'un informaticien, un informaticien est avant tout un informaticien, et il existe une certaine somme de " connaissances élémentaires " nécessaire à sa formation.

Sur ces " connaissances élémentaires " reposent toutes les suites possibles, dépendantes, cette fois, d'une spécialisation.

- Le projet OEDIPE est un projet d'enseignement.

La finalité de cet enseignement, c'est l'acquisition de ces " connaissances élémentaires ".

Quelles sont ces " connaissances élémentaires " ?

Elles sont encore à définir avec précision :
Il n'est possible, pour l'instant, que de proposer des " têtes de chapitres " :

- Structure d'un ordinateur
- Fonctionnement d'un ordinateur
- Utilisation d'un ordinateur
  - . les langages.
  - . les commandes.
- Programmation d'un ordinateur
  - . les structures de données.
  - . l'algorithmique.
  - . le choix d'un langage.
- Choix d'un ordinateur
  - . les critères de différenciation.
  - . les critères de prévision.

b) L'ensemble des notions

Si la globalité des " connaissances élémentaires " est encore à définir avec précision, il est évidemment difficile de citer toutes les notions nécessaires au passage entre " l'avant-cours " et la connaissance.

• En ce qui concerne les trois premiers volets (structure, fonctionnement et utilisation d'un ordinateur), le système SCALP (décrit dans le chapitre suivant) prétend résoudre, en partie, les problèmes.

• Il reste les deux derniers volets à analyser.

Ce qu'il est possible d'affirmer :

• sur les structures de données et l'algorithmique :

Il ne s'agit pas d'enseigner quelques exemples, mais d'utiliser ces quelques exemples pour enseigner des méthodes de recherche et d'analyse.

• sur le choix d'un langage :

Il s'agit d'enseigner des méthodes de choix - à partir des exemples, schématiquement décrits, que sont les langages actuels.

Ces méthodes de choix peuvent conduire à des méthodes sommaires d'élaboration de nouveaux langages.

• sur le choix d'un ordinateur :

Il s'agit, d'une part, de montrer que les ordinateurs utilisés pour la pratique du cours ne sont pas les seuls au monde, et de montrer que tous les ordinateurs actuels seront, relativement vite, dépassés.

Il s'agit, d'autre part, d'enseigner des méthodes d'analyse critique d'un ordinateur quelconque.

c) La démarche pédagogique :

. Déterminer les notions à enseigner n'est que la base d'un enseignement.

. Etablir un ordre logique d'enseignement de ces notions à partir de leur analyse n'est que l'ébauche d'un enseignement.

. Pour passer de l'ébauche d'un enseignement à cet enseignement, il reste un dernier travail à accomplir (la réalisation sera alors possible).

Ce dernier travail est l'étude de ce que j'appelle " une démarche pédagogique ".

Une démarche pédagogique se définit par :

- Une conception précise d'un processus d'acquisition des connaissances (par exemple : sensibilisation - adaptation - acquisition - renforcement).

- Une connaissance précise des outils utilisés.
  . soit des outils existants (maîtres, livres, devoirs, machines à enseigner, ensembles audio-visuels,...).
  . soit des outils spécialement construits.

- Une compréhension précise des rapports notions-outils-élèves.

- La réalisation sera alors possible : il ne s'agit que d'un travail technique de mise en oeuvre.

CHAPITRE II

LE SYSTEME SCALP

PRESENTATION DU SYSTEME

- Finalités et durée du cours
- Les outils utilisés
- La population enseignée

DESCRIPTION DU SYSTEME

- Analyse diachronique
- Analyse synchronique

SCHEMAS SYNOPTIQUES

EVALUATION DU COUT

SUBSEQUENCE

II § 1 - PRESENTATION DU SYSTEME SCALP -

§ 1.1. - FINALITE DU COURS :

Le cours donne :

. une connaissance-pratique-de l'organisation fondamentale des ordinateurs.

- le travail pratique est effectué sur un ordinateur particulier (l'expérimentation a été faite sur le P.D.P.8 de la Digital Equipment Corporation - cf chapitre 3)
- Mais le cours n'enseigne pas cet ordinateur. Il considère cet ordinateur comme un exemple d'une part, et comme un outil d'autre part.

. une connaissance-pratique-des premiers éléments de la programmation.:

- L'algorithmique, en tant que telle, n'est pas abordée ; mais elle est présentée comme conséquence de la nature de l'ordinateur.

- Les concepts acquis à la fin du cours :

. La nature et le rôle de la mémoire.
. La notion d'adresse.
. La notion d'instruction (donc d'opérateur et d'opérande).
. Les problèmes de l'adressage.
. La notion d'exécution d'une instruction.
. Le principe de l'exécution séquentielle (ainsi que des tests et des aiguillages).
. La notion de programme et de sous programme.
. La notion de transfert d'informations.
. Le principe des interruptions.
. La nature et le rôle d'un software de base (assembleur, chargeur, langage de commande).

§ 1.2. - DUREE DU COURS.

. L'élève a deux tâches :
  - un travail individuel.
  - un travail collectif (maître et groupe d'élèves).

. Seul le temps nécessaire au travail collectif peut être précisément évalué.
  - environ 24 heures.

. Quant au travail individuel, la durée dépend de chacun.
  - il faut compter, en moyenne, un tiers du temps consacré au travail collectif, soit, environ, 8 heures.

. Donc, le temps-complet-à consacrer à cet enseignement est une trentaine d'heures.

§ 1.3. - LES " OUTILS " UTILISES :

. Un maître : libéré d'une part de sa tâche purement didactique, il peut se consacrer, essentiellement, à un travail de guide et de conseiller.

- la partie " mécanique " de l'enseignement est en effet automatisée et le " programme " d'enseignement est parfaitement défini par l'analyse effectuée selon les principes de l'enseignement assisté.

. Un ensemble " audio-visuel " : il assure la concrétisation (validation) de tout fait énoncé lors d'un travail collectif " hors machine ".

. Différents manuels : enseignement programmé, schémas, tableaux synoptiques, textes succints.

Ils réduisent la durée du travail théorique individuel, tout en augmentant sa rentabilité (mémorisation rapide, mais précision en tout domaine étudié).

. Un ensemble d'enseignements assistés par ordinateur :

il permet un travail pratique individuel et efficace puisque ce travail est contrôlé et immédiatement corrigé (automatiquement).

. Et, enfin, l'ordinateur lui-même :

sa pratique assure la " réalité " des faits enseignés, sur un cas particulier ; ce cas particulier étant généralisé lors du travail théorique.

§ 1.4. - LA POPULATION ENSEIGNEE :

Le cours peut s'adresser à trois types de population (dans les 3 cas, cette population, au moins bachelière, n'a pas encore fait de programmation).

. Une population de futurs informaticiens :

- par exemple :
  - les étudiants de première année d'une maîtrise d'informatique.
  - les étudiants de première année d'une école d'ingénieurs.
  - les étudiants de première année d'un institut universitaire de technologie.
- ce cours, alors, s'intègre à un enseignement annuel plus général (cet enseignement auquel s'intéresse le projet OEDIPE).
- les élèves sont, dans ce premier cas, des universitaires. (leur motivation est souvent imprécise).

. Une population de spécialistes en certains domaines qui s'intéressent à l'informatique pour, au besoin, l'utiliser.

- par exemple : des chimistes, des physiciens, des linguistes, des mathématiciens.
- ce cours, là encore, s'intègre à un enseignement (annuel ?) plus général. C'est d'ailleurs, peut-être, le même enseignement que celui précédemment cité.

(les élèves sont, cette fois, précisemment motivés).

. Une population de spécialistes en certains domaines qui ont besoin de connaître l'ordinateur sans avoir à l'utiliser.

- par exemple : des philosophes, des sociologues, des psychologues, des littéraires qui estiment difficile de parler de la société contemporaine sans connaître, globalement, l'ordinateur.
- le cours est alors autonome. La connaissance diffusée est suffisante à la démythification de l'ordinateur.

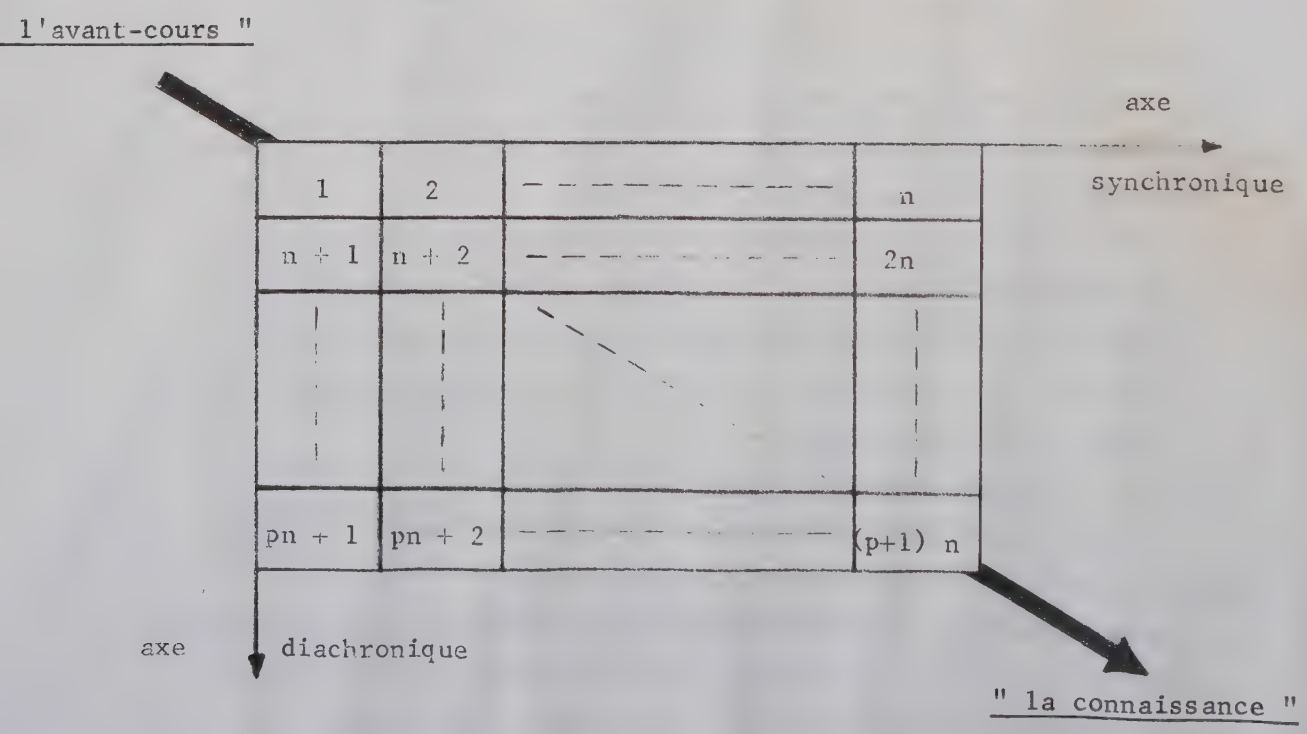
II § 2 - DESCRIPTION DU SYSTEME SCALP -

INTRODUCTION : Il faut maintenant décrire, avec précision, ce système d'enseignement.

Nous ne reviendrons pas, dans cette description, sur la durée du cours, et sur la population enseignée (ces deux critères sont définis dans le paragraphe précédent).

Nous nous attacherons, essentiellement, aux " moyens " d'atteindre la finalité décrite.

- Deux composantes de ces " moyens " sont à analyser :
  - . La première, c'est la succession des notions à enseigner pour effectuer le passage entre " l'avant-cours " et la connaissance (c'est ce que j'appelle la diachronie).
  - . La seconde, c'est, pour un groupe de notions déterminé, la succession des outils et méthodes utilisés pour effectuer le passage entre la sensibilisation aux notions et la rétention de ces notions.
    (c'est ce que j'appelle la synchronie).



II § 2.1 - ANALYSE DIACHRONIQUE -

C'est une partition ordonnée par le temps d'apprentissage de l'ensemble des notions enseignées.

Le cours présente vingt centres d'intérêt.

1°) - LE CALCUL AUTOMATIQUE :

. Pour un problème donné, on détermine un algorithme qui permet d'expliquer un principe de calcul automatique (le calcul utilise la syntaxe d'écriture de l'algorithme, et non la sémantique).

Cf Figure II.1.

. notion de machine :

C'est un " automate " qui effectue un calcul défini par un certain algorithme sur certaines données.

. notion d'information :

C'est, et l'algorithme, et les données.

. le travail d'un utilisateur :

a) . déterminer l'algorithme (analyse du problème ; résolution formelle du problème).
. fournir l'algorithme à l'automate.

b) . fournir les données à l'automate.
. activer l'automate.
. récupérer les résultats.

N. B. : Le travail a est long mais unique.
Le travail b est très court et peut se répéter, sans inconvénient, pour tout nouveau jeu de données (l'algorithme restant utilisable).

Cf Figure II. 2.

2°) - STRUCTURE GLOBALE D'UN ORDINATEUR.

. Organe de support de l'information.

- contient l'algorithme.
- contient les données.
- contient les résultats.

. Organe de traitement de l'information.

- décrit l'algorithme en traitant les " paramètres effectifs " (données).
- détermine les résultats.

. Organes de transferts de l'information.

- un organe d'entrée de l'information, permet de placer l'algorithme d'une part, les données d'autre part, dans le support de l'information.
- un organe de sortie de l'information permet d'extraire les résultats du support de l'information.

. Organe de commandes.

- permet d'activer l'ordinateur.

Cf. Figure II. 3.

3°) - DESCRIPTION DE L'UNITE MEMOIRE.

. Information élémentaire - mémoire élémentaire - le bit.
. La mémoire centrale composée de N bits, donc de N informations élémentaires.
. Information fondamentale - mémoire fondamentale le " mot-mémoire " (n bits).
. La mémoire centrale est composée de N/n mots-mémoire.
. Repérage d'une information fondamentale - l'adresse d'un mot-mémoire.

. Les registres de l'unité mémoire :

- un registre " adresse " (R.A. : contient l'adresse d'un mot pour sa sélection).
- un registre " tampon " (R.T. : contient une information à " écrire " en mémoire, ou reçoit une information " lue " en mémoire).

. La gestion de l'information.

- les circuits de sélection (RA → Mémoire centrale).
- les circuits de lecture (Mémoire centrale → RT).
- les circuits d'écriture (RT → Mémoire centrale).

. Le rôle de l'unité mémoire.

- situation par rapport au schéma global d'un ordinateur.

. Remarque : notion de mémoire banalisée.

- aucun sens n'a été donné à l'information.

Cf. Figure II. 4.

4°) - DESCRIPTION DE L'UNITE DE TRAITEMENT.(processeur).

. Notion de codage de l'information - langage machine.
. Notion d'instruction (type d'information).
. Exécution d'une instruction (décodage - activation de circuits).
. Notion de programme (ensemble ordonné d'instructions).
. Exécution d'un programme - (registres et circuits le permettant).

- rappel : les registres de l'unité mémoire RA et RT.
  . lieu entre le processeur et la mémoire.
- un compteur ordinal C O.
  . détermine l'instruction à exécuter après l'instruction courante.
- un registre instruction R I.
  . contient le type d'opération à effectuer pendant tout le travail nécessaire à l'exécution de l'instruction.

- les registres modifiables par instructions R M.
  . enregistrant les résultats des opérations logiques et arithmétiques.
- les circuits du processeur :
  . les circuits permettant l'exécution des instructions.
  . les circuits permettant l'enchaînement des travaux.
- les éléments de synchronisation.

. Organigramme global de traitement.

Cf. Figure II. 5 et Figure II. 6.

5°) - DESCRIPTION DU PUPITRE DE CONTROLE ET DE COMMANDES.

. Visualisation des registres (lampe-bit).
. Interrupteurs d'initialisation, de contrôle et de commandes :

- entrée manuelle d'informations.
- examen mémoire.
- activation de l'ordinateur.
- arrêt manuel de l'ordinateur.
- mode d'exécution (direct, pas à pas).

6°) - ETUDE DES INSTRUCTIONS A OPERANDES IMPLICITES.

- C'est un cas simple de fonctionnement de l'ordinateur : les problèmes de l'adressage ne sont pas encore abordés.

. Les instructions - ne traitant que les registres R M.
(Opérations arithmétiques, complémentations, décalages, tests).

. Instruction de fin de programme.

7°) - ETUDE CONCRETE DU FONCTIONNEMENT DE L'ORDINATEUR AVEC DES PROGRAMMES COMPOSES D'INSTRUCTIONS A OPERANDES IMPLICITES.

Des précisions sont données à l'organigramme du § 4.
.Exemples.

8°) - ETUDE DES PROBLEMES DE L'ADRESSAGE.

. Généralités sur les instructions à opérande explicite :
- code opération-information complémentaire-indications sur les opérandes en mémoires.

. Détermination de l'adresse d'un opérande en mémoire.
- pagination.
- adressage indirect.
- adressage indexé.
- adressage " immédiat ".

. Détermination de l'opérande en mémoire.
- connaissant l'adresse d'un opérande, il suffit de faire une lecture en mémoire pour obtenir l'opérande.

9°) - LES " ETATS " DU PROCESSEUR.

On entend par " état du processeur " ce qui correspond à un transfert " mémoire - processeur " :
- lecture en mémoire des instructions.
- lecture en mémoire des adresses " indirectes ".
- lecture en mémoire des opérandes.
- écriture en mémoire de valeurs.

- Les différents états sont semblables :
- transfert d'information.
- traitement de l'information.

L'état " spécifie " la nature exacte de l'information.

- Etude de la coordination des différents états.

10°) - ETUDE DES INSTRUCTIONS A OPERANDES EXPLICITES.

- opérations logiques et arithmétiques.
- rangement et chargement.
- branchements et tests.

11°) - ETUDE CONCRETE DU FONCTIONNEMENT DE L'ORDINATEUR AVEC DES PROGRAMMES COMPOSES D'INSTRUCTIONS A OPERANDES EXPLICITES OU IMPLICITES.

- organigramme précis, et complet, du fonctionnement.
- exemples.

12°) - ETUDE DE LA PROGRAMMATION EN LANGAGE MACHINE.

. Transcription d'un algorithme en langage " symbolique ".
. Transcription d'un langage " symbolique " en langage binaire.
(assemblage manuel).
. Chargement de l'ordinateur par le pupitre de contrôle et de commandes.
. Activation de l'ordinateur par les interrupteurs de commandes.
. Lecture des résultats par le pupitre de contrôle.

13°) - ETUDE D'UN EXEMPLE D'ENTREES/SORTIES D'INFORMATIONS.
- Etude réalisée sur un exemple simple : un téléscripteur.

. Les problèmes de codage des caractères.
. Les problèmes de temps dans les transmissions
(besoin de convertisseurs séries ↔ parallèles).
. Les unités de contrôle du téléscripteur.
- un registre sas (buffer) pour stocker l'information.
- un indicateur (flag) qui permet au processeur de reconnaitre.
quand une transmission est possible.
. Les instructions de transfert : téléscripteur ↔ ordinateur.
- instructions de tests du " flag ".
- instructions de transfert entre le processeur et le " buffer ".

14°) - ETUDE DES INTERRUPTIONS.

. Principe des interruptions.
. Mode où les interruptions sont possibles.
    - traitement des interruptions.
. Mode où les interruptions sont masquées.
. Les instructions permettant, ou masquant, les interruptions.

15°) - ETUDE CONCRETE DU FONCTIONNEMENT DE L'ORDINATEUR AVEC L'ENSEMBLE DES INSTRUCTIONS ETUDIEES.

- Ce qu'apportent les instructions de transfert.
- Exemples.

Cf. Figure II. 7.

16°) - ETUDE DE LA PROGRAMMATION EN LANGAGE NUMERIQUE LN.
(Hexadécimal, octal, ...).

. Ecriture en langage machine et chargement d'un assembleur-chargeur LN.
(téléscripteur → ordinateur).
. Ecriture en langage machine et chargement d'un programme de dump LN.
(ordinateur → téléscripteur).
. Ecriture en langage machine et chargement d'un programme de commandes
(par téléscripteur).

. Transcription d'un algorithme en langage " symbolique ".
. Transcription du langage " symbolique " en langage LN.
(assemblage manuel).
. Assemblage et chargement par l'assembleur-chargeur LN.
. Activation de l'ordinateur par le programme de commandes.
(Y compris l'arrêt par interruption).
. Lecture des résultats par la demande d'un dump.

17°) - ETUDE SUCCINTE DES ORGANES DE TRANSFERTS.

. Tambours, disques, bandes magnétiques.
. Terminaux graphiques.
. Principe d'un interface.

18°) - ETUDE D'UN SOFTWARE DE BASE.

. Principe d'un assembleur.
. Principe d'un chargeur.
. Principe d'un langage de commandes.

19°) - ETUDE DE LA PROGRAMMATION EN LANGAGE SYMBOLIQUE. LS.

. Ecriture d'un algorithme.
. Transcription de cet algorithme dans le langage symbolique propre à l'assembleur utilisé. (LS)
. Edition du programme.
. Assemblage du programme.
. Chargement du programme.
. Exécution du programme.

20°) - EXTENSIONS POSSIBLES (Conclusion au cours).

. Description plus étendue des systèmes possibles (conversationnel-temps partagé ...).
. Aperçu sur les principes généraux de la programmation.

FIGURE II.1.

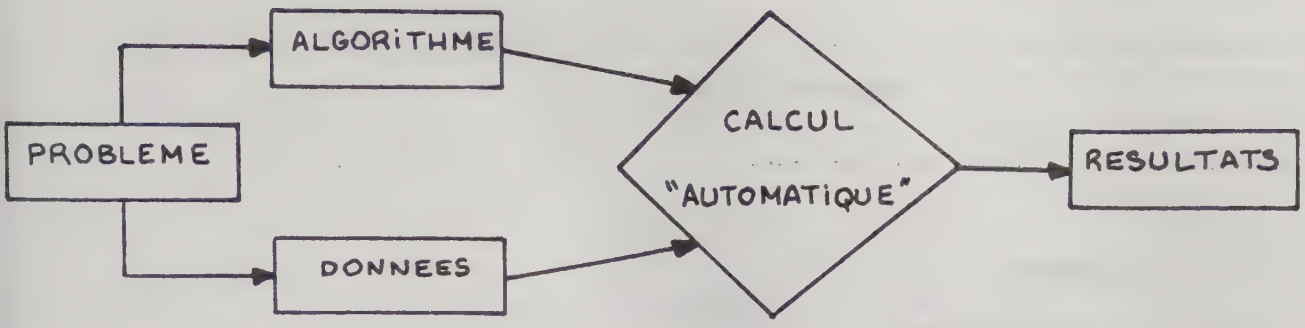


FIGURE II.2.

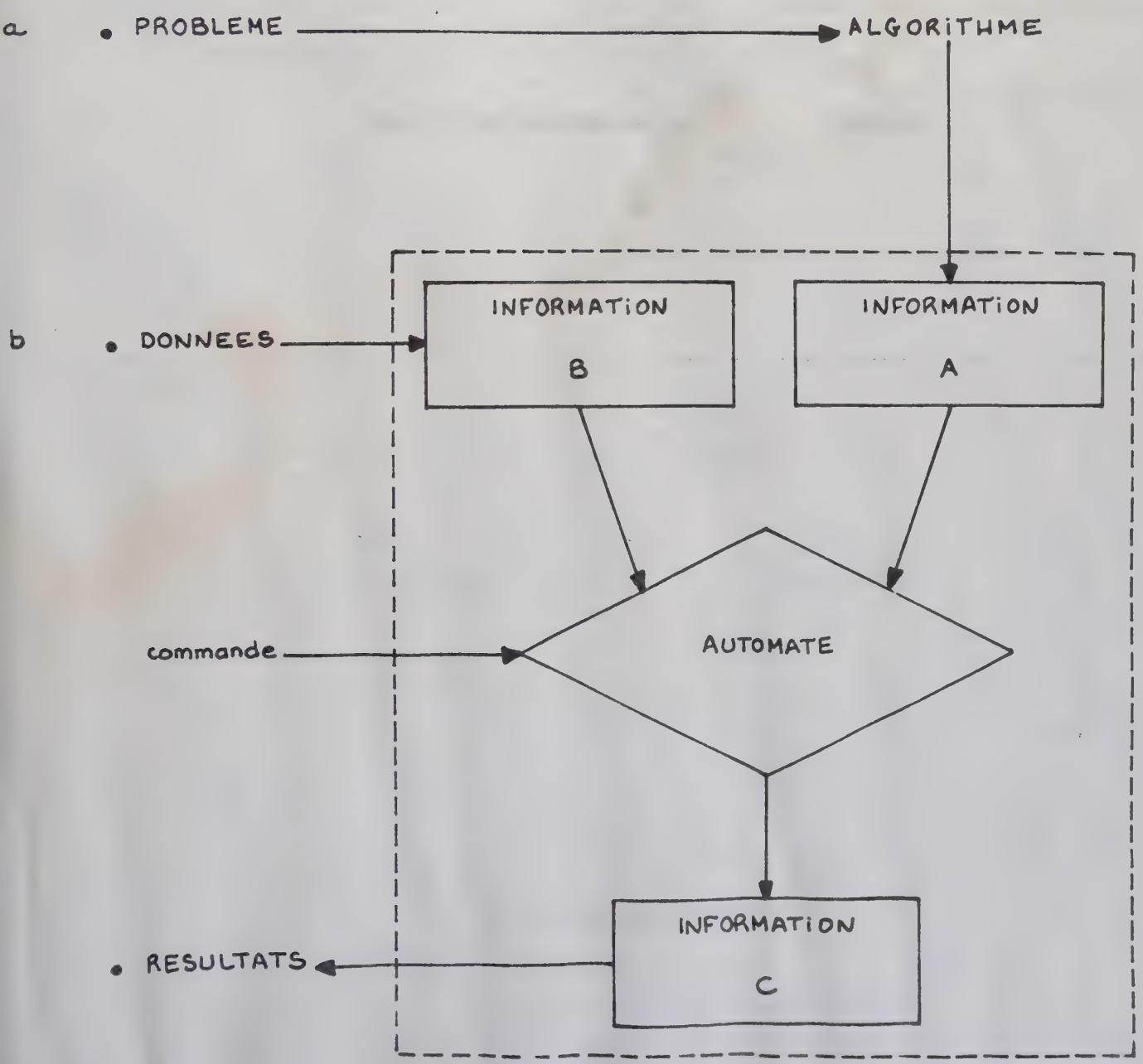


FIGURE II.3.

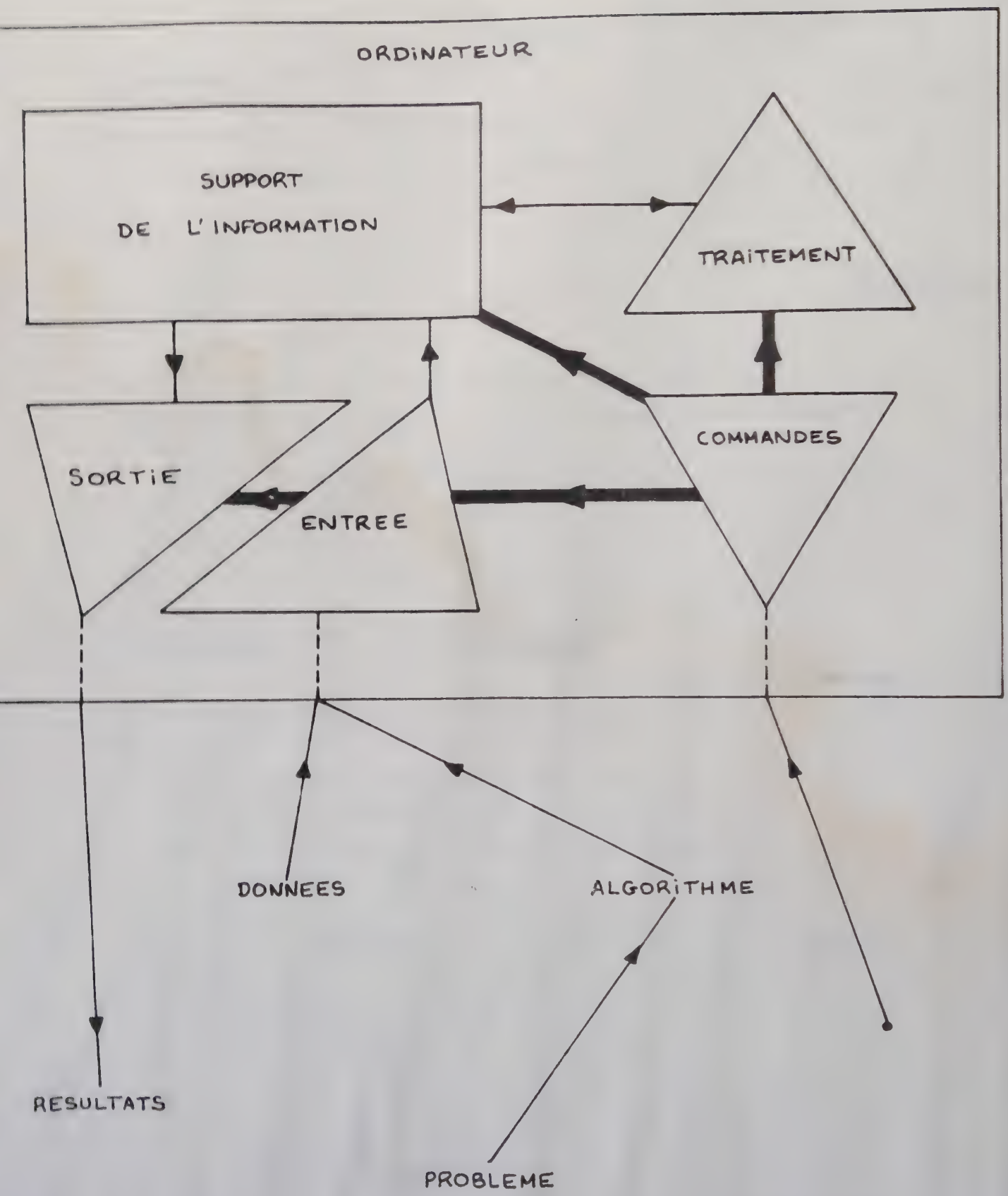

ORDINATEUR
SUPPORT
DE L'INFORMATION
TRAITEMENT
SORTIE
ENTREE
COMMANDES
DONNEES
ALGORITHME
RESULTATS
PROBLEME

FIGURE II.4

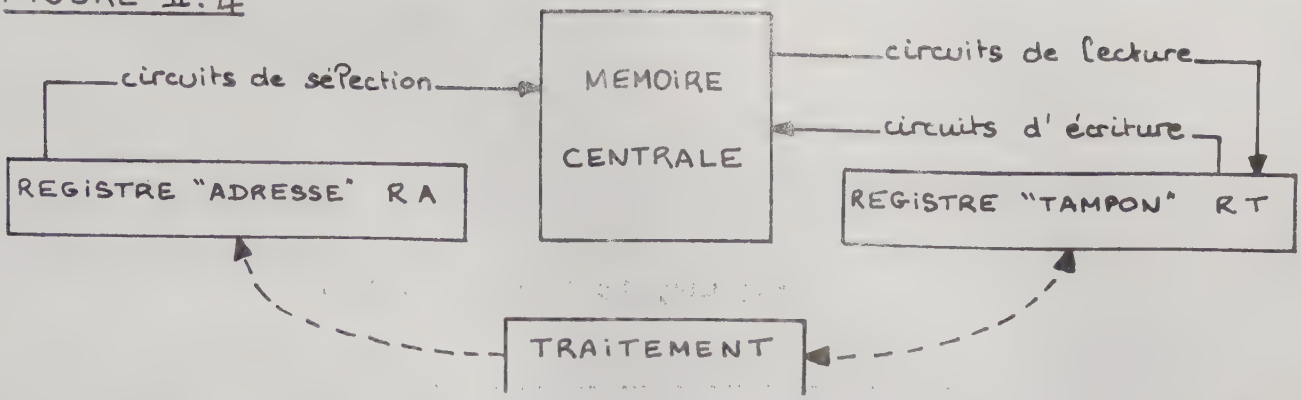


FIGURE II.5

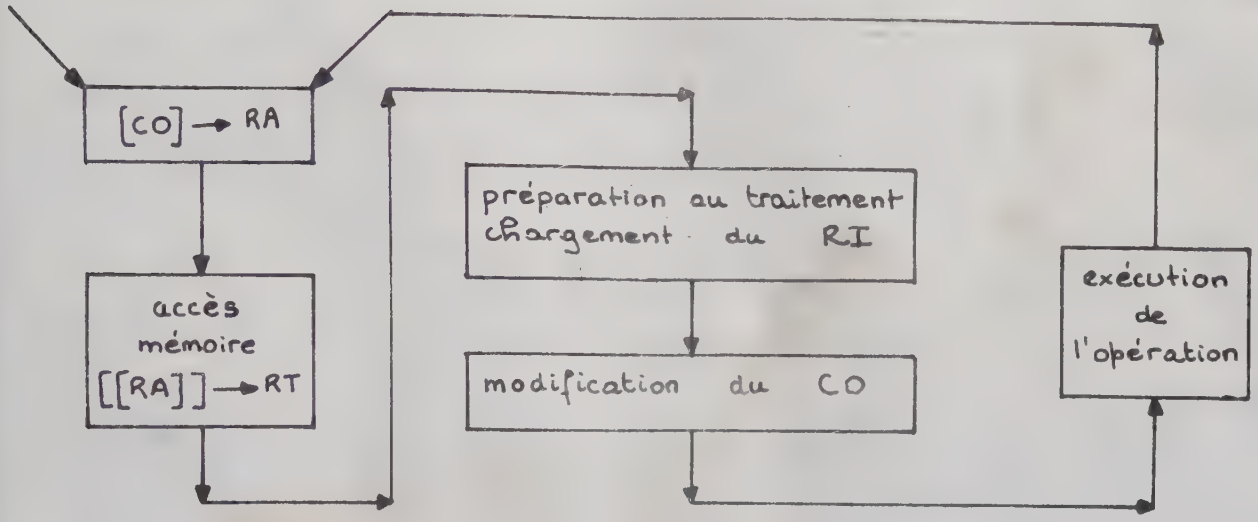


FIGURE II.6

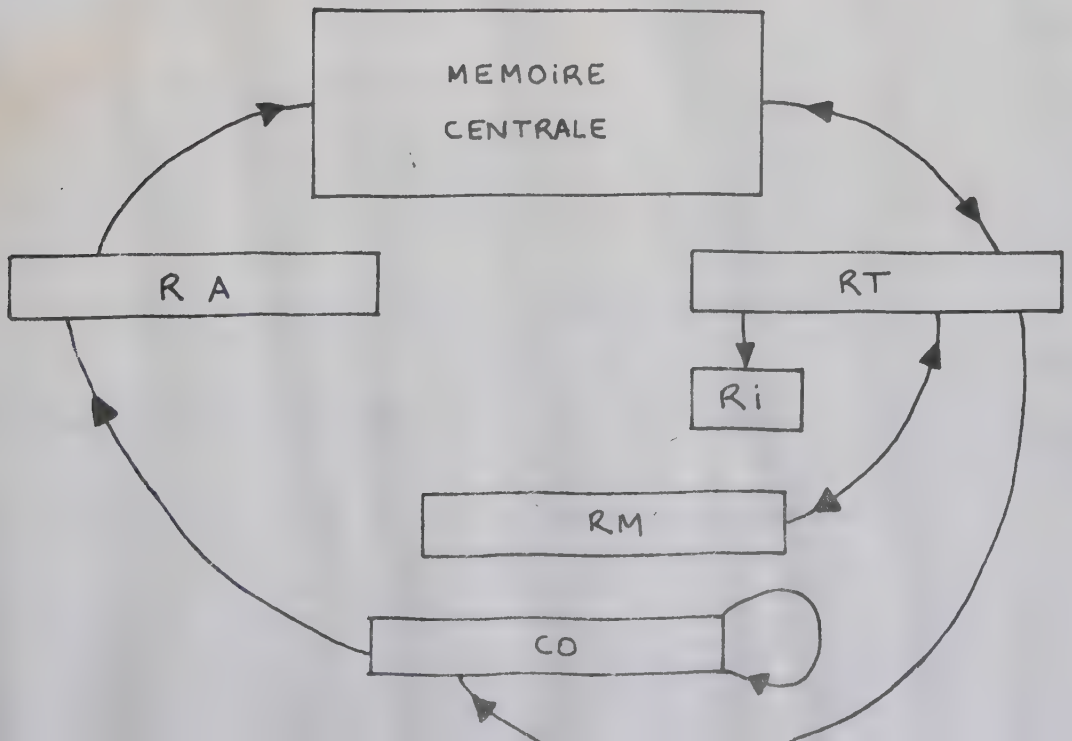
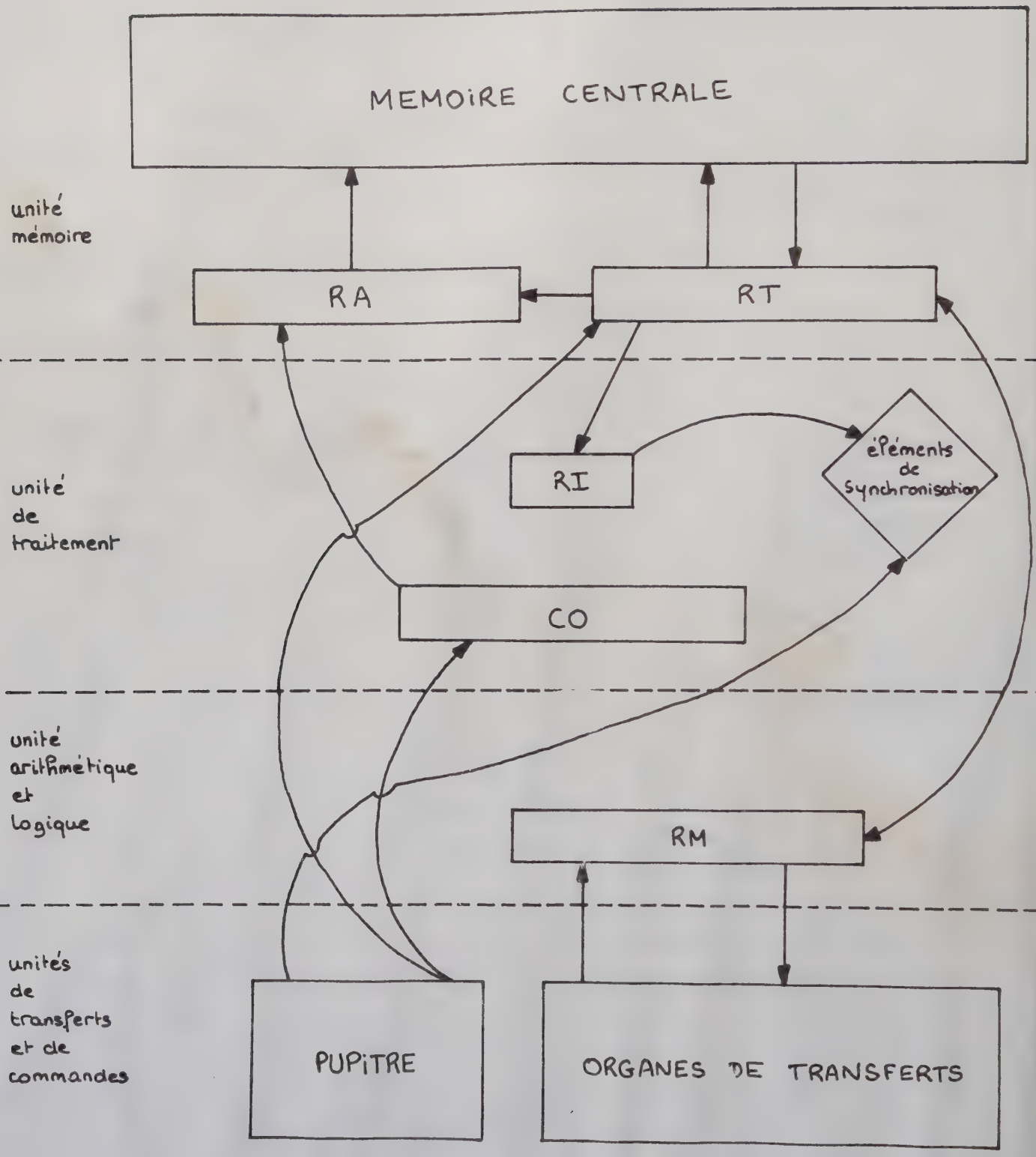


FIGURE II.7


MEMOIRE CENTRALE
unité mémoire
RA
RT
RI
éléments de synchronisation
unité de traitement
CO
unité arithmétique et logique
RM
unités de transferts et de commandes
PUPITRE
ORGANES DE TRANSFERTS

II § 2. 2 - ANALYSE SYNCHRONIQUE -

C'est la démarche pédagogique suivie.

L'enseignement est donné par cycles (correspondant à l'axe diachronique) de quatre temps pédagogiques :

T1 : Ce premier temps est le temps de sensibilisation aux notions nouvelles. Cette sensibilisation doit susciter, chez l'élève, la découverte des analogies entre ses connaissances antérieures et les nouveaux concepts.

T2 : Ce deuxième temps est le temps d'adaptation aux notions suggérées dans le premier temps : constatation des faits globaux ; premières expérimentations collectives.

T3 : Ce troisième temps est le temps d'acquisition, par la manipulation, des notions développées dans les premier et deuxième temps.
Ce troisième temps est le temps pédagogique essentiel.

T4 : Ce quatrième temps est le temps de synthèse. C'est le contrôle et le renforcement des acquisations en vue de la coordination et de la rétention des connaissances. L'enseignement conçu étant essentiellement analytique, il est nécessaire de situer chaque notion dans son contexte à l'intérieur de son cycle, ainsi que de situer chaque cycle à l'intérieur de l'ensemble du cours.

1°) - TEMPS DE SENSIBILISATION.

Ce premier temps est une sorte de préambule.

Au moment d'aborder un nouveau cycle d'enseignement, l'élève a un certain acquis (antérieur au cours ou donné par les cycles précédents).

Pour mieux comprendre les notions nouvelles apportées par ce nouveau cycle, il est important, avant de les aborder, de lui faire pressentir l'existence de ces notions nouvelles.

L'acquis permet ainsi de poser des problèmes nouveaux dont la résolution fait l'objet des temps pédagogiques suivants.

- acquis → problèmes nouveaux.
- notions nouvelles ↔ résolution de ces problèmes.

Le but de ce temps de sensibilisation est de permettre le passage de l'acquis aux problèmes nouveaux. (Nous remarquons alors que l'élève a déjà réalisé, intuitivement, une synthèse globale entre l'acquis et les notions nouvelles non encore explicitées).

Ce premier " enseignement " est donné par un manuel (M1) rédigé sous forme d'enseignement programmé (présentation-question-choix multiple de réponses - aiguillage).

M1 a six chapitres :

Chaque chapitre peut être divisé en deux parties :

- la première est très brève : c'est une révision de l'acquis pour lui apporter une nouvelle structuration. Cette nouvelle structuration prépare l'énoncé des problèmes nouveaux.
- la seconde est la plus importante : c'est l'énoncé, progressif, au rythme de l'élève, de ces problèmes nouveaux.

M 1 (1) : Premier chapitre.

. L'acquis, c'est la connaissance antérieure au cours.

. La " révision " met en évidence les notions de problème, de données, de processus de calcul, de résultats, de fonction de mémoire, d'automate.

Cette " révision " permet d'aborder un premier problème :

. - Le calcul automatique.

Ce problème étant posé, il est possible d'aborder un second problème :

. - La création d'un " automate " puisant de l'information et la traitant.

Ce premier chapitre correspond aux § 1 et 2 de l'analyse diachronique.

M 1 (2) : Deuxième chapitre.

. Le nouvel acquis, c'est la structure globale d'un ordinateur (support d'informations, traitement d'informations, transfert d'informations) et le rôle et la nature de l'unité mémoire de l'ordinateur (mots, adresses, contenus ; sélection, lecture, écriture).

. Une révision met en évidence la notion de mémoire banalisée (et l'algorithme et les données sont représentés dans le même langage sans distinction).

Cette révision permet d'aborder un premier problème :

. - les différents types d'information - leur distinction.

Puis un second problème :

. - le traitement d'un ensemble ordonné d'informations.

Enfin un troisième problème :

. - le rôle d'un manipulateur.

Ce second chapitre correspond aux § 4 et 5 de l'analyse diachronique.

M 1 (3) : Troisième chapitre.

. Le nouvel acquis, c'est le rôle et la nature de l'unité de traitement de l'ordinateur (exécution globale d'un programme), le rôle et la nature du pupitre de contrôle et de commandes, un premier jeu d'instructions - les instructions à opérandes implicites).

. Une révision montre que les seules instructions connues ne modifient en rien la mémoire et permet d'aborder un premier problème.:

. - nécessité de traiter des valeurs en mémoires (opérandes).

Donc un second problème :

. - nécessité de " lire " ces valeurs

Enfin, un troisième problème :

. - comment coder une instruction à opérande en mémoire

(les problèmes de l'adressage).

Ce troisième chapitre correspond aux § 8 et 9 de l'analyse diachronique.

M 1 (4) : Quatrième chapitre.

. Le nouvel acquis, c'est l'adressage de l'ordinateur,

- les différents " états " du processeur.
- un second jeu d'instructions

(opérandes explicites)

. Une révision montre que l'ordinateur peut traiter un certain nombre d'opérations et qu'un algorithme devra être écrit avec ces opérations.

Un premier problème se pose :

. - Comment écrire tout algorithme avec ce minimum d'opérations.

Puis un second problème :

. - Un algorithme étant écrit en langage machine, quel est le travail à accomplir pour en réaliser l'exécution.

Ce quatrième chapitre correspond au § 12 de l'analyse diachronique.

M 1 (5) : Cinquième chapitre.

. Le nouvel acquis, c'est l'écriture d'algorithme en langage machine et c'est la méthode de chargement-exécution par le pupitre de contrôle et de commandes.

. Une révision montre les difficultés que présente la seule communication pupitre → ordinateur.

Un premier problème se pose :

. - est-il possible d'utiliser un autre moyen de transferts d'informations.

Puis un second problème :

. - un " périphérique " quelconque relié à l'ordinateur ne ralentira-t-il pas le fonctionnement de cet ordinateur.

Puis un troisième problème :

. - serait-il possible, alors, de programmer autrement qu'en binaire.

Ce cinquième chapitre correspond aux § 13 et 14 de l'analyse diachronique.

M 1 (6) : Sixième chapitre.

. Le nouvel acquis, ce sont les entrées/sorties à l'aide du téléscripteur, les interruptions, une programmation en numération non binaire LN à l'aide du téléscripteur.

. Une révision montre que si le langage LN est plus simple à manier que le langage binaire, les problèmes de la programmation numérique sont les mêmes.

Un problème se pose :

. - est-il possible, avec les moyens que l'on a, d'écrire un programme en langage symbolique. Un programme (écrit numériquement, une fois pour toute) traduirait ce symbolique en binaire).

Ce sixième chapitre correspond au § 18 de l'analyse diachronique.

2°) - TEMPS D'ADAPTATION.

Ce deuxième temps est un temps de dialogue.

Il faut énoncer précisément les problèmes nouveaux, en débattre, et les résoudre.

Ce second " enseignement " met en présence :

- le maître (qui connait les " notions nouvelles ", donc la résolution effective des problèmes).
- un groupe d'élèves (chaque élève étant sensibilisé, à sa manière, aux problèmes nouveaux).

. Le maître expose avec précision les problèmes.

. Le dialogue peut s'ouvrir

- dialogue 1 : maître ↔ élève.
- dialogue 2 : élève ↔ élève.

. Le maître - et les élèves - critiquent les " solutions " exposées.

. Le maître propose la solution propre à l'ordinateur enseigné (en insistant sur le fait qu'il ne s'agit que d'un cas particulier : parmi les solutions exposées, certaines peuvent être cohérentes mais n'ont pas été adoptées par les constructeurs).

Cet " enseignement " est assisté par moyens audio-visuels.
Le moyen audio-visuel le plus souple et le plus complet est certainement, pour cet enseignement, le magnétoscope (enregistreur d'images et de sons), une caméra de télévision, et un circuit fermé de téléviseurs.

N.B. : le magnétoscope doit permettre l'arrêt sur l'image.

. Réalisation de l'audio-visuel : avant la séance, le maître filme les cas exemplaires de fonctionnement de l'ordinateur.

Essentiellement, l'image présentera le pupitre de contrôle (donc de visualisation) et de commandes, et son utilisation par un manipulateur.

Le manipulateur charge un programme (ou bien le programme est déjà chargé) et en commande l'exécution (directe ou pas à pas).

Le manipulateur enregistre les commentaires sur toutes les opérations qu'il effectue.

. Utilisation de l'audio-visuel :

Lorsque le maître a terminé son exposé sur les solutions adoptées par les constructeurs, il concrétise ces notions nouvelles par la projection de ce film commenté.

L'élève vérifie la possibilité et la réalité de ces notions nouvelles par un contact, assez proche, avec l'ordinateur.

3°) - LE TEMPS D'ACQUISITION -

Ce troisième temps est le temps pédagogique essentiel.
Nous l'avons dit, la programmation est essentiellement manipulatoire.

Après l'énoncé des problèmes nouveaux, après l'examen des différentes solutions envisagées, après l'énoncé des solutions propres à l'ordinateur, l'élève doit manipuler les notions nouvelles pour enfin les acquérir.

Ce troisième " enseignement " est donc un enseignement pratique.

Quel système l'élève va-t-il utiliser ?

. Un système créé par le constructeur pour l'utilisation complète de l'ordinateur ?

- Ceci est impossible.

- Ce système n'a pas été conçu pour l'enseignement (une utilisation fonctionnelle n'est pas nécessairement pédagogique).

- Il est difficile, dans un tel système, d'isoler les notions connues ou à acquérir dans l'immédiat des autres notions non encore étudiées.

. L'ordinateur lui-même ?

- Ceci, aussi, est impossible.

- Le travail pratique doit être individuel : or l'ordinateur est unique et les élèves sont nombreux.

- L'ordinateur choisi est " petit " et son fonctionnement est "simple ". Mais il est encore trop complexe pour une grande partie du cours.

. Un ensemble de systèmes conçu pour l'élève ?

- C'est la solution adoptée.

Conception de ces systèmes (phases) :

. Ces phases sont conversationnelles et en temps partagé.
  - le travail est individuel (même au sein d'un travail collectif).

. Ces phases sont progressives :
  - chaque phase n'utilise que les connaissances envisagées.
  - pour un nouvel acquis, une autre phase est utilisée.

. Le travail de l'élève :
  - l'élève travaille, seul, sur son téléscripteur.
    . Son travail est indépendant de ce qui l'entoure.
  - Un groupe d'élèves travaille, en même temps, dans une salle de téléscripteurs.
    . Le dialogue élève ↔ élève est possible - et très profitable. (les travaux demandés, portant sur des données aléatoires, ont donc une forme différente. Le dialogue, ainsi, ne peut porter que sur le fond et c'est donc une prolongation du dialogue entamé dans le deuxième temps pédagogique).
  - Le maître est présent pour aider chaque élève, en particulier.
  - L'élève dispose d'un manuel (M 2) qui lui présente, pour chaque séance, le travail à accomplir, et lui précise la finalité de ce travail.

. Cinq phases sont ainsi programmées :

  - P 1 : les différents systèmes de numération utilisés.
  - P 2. 1 : fonctionnement global de l'ordinateur.
  - P 2. 2 : fonctionnement précis de l'ordinateur.
  - P 3 : utilisation (numérique) de l'ordinateur.
  - P 4 : utilisation (symbolique) de l'ordinateur.

P 2 (1 et 2), P 3 et P 4 utilisent une simulation de la partie [illegible]gée de l'[illegible]dinateur. Nous l'appellerons " ordinateur-virtuel ".

. L'ordinateur est utilisé.

. Une séance a lieu devant l'ordinateur lui-même par groupes de travail (divisés en équipes).

- Cette séance permet de présenter - et c'est nécessaire - l'ordinateur.

- Cette séance est utilisée pour la manipulation des systèmes d'entrées/sorties (étudiées donc en temps réel).

. Les " entrées-sorties " ne sont pas simulées dans " l'ordinateur-virtuel ".
P 2 (1 et 2) est utilisée avant l'enseignement des entrées-sorties.
P 3 aussi, et la simulation des entrées-sorties rendrait confus l'emploi du téléscripteur (à la fois téléscripteur et " ordinateur-virtuel ").
P 4 est utilisée après, mais essentiellement pour une introduction au software de base : l'adjonction des entrées-sorties n'ajouterait rien à la phase.

P 1 : PHASE NUMER

- Cette phase propose à l'élève une manipulation de la numération binaire et d'une numération " LN " (hexadécimal, octal,...).

Cette manipulation est décomposée en chapitres traitant les centres d'intérêt principaux.

Chaque chapitre est décomposé en paragraphes.

Chaque paragraphe propose un certain nombre n d'exercices. C'est-à-dire que n exercices doivent être faits correctement - les exercices faux ne sont pas comptés.

- Description :

. Chapitre I - Conversions binaire ↔ LN ↔ décimal ↔ binaire.

I. 1 : 2 exercices de conversions décimal → binaire.
I. 2 : 2 exercices de conversions binaire → LN.
I. 3 : 2 exercices de conversions LN → binaire.
I. 4 : 2 exercices de conversions LN → décimal.

. Chapitre II - Opérations en numération binaire.

II. 1 : 2 exercices d'addition.
II. 2 : 2 exercices de " double d'un nombre " - (Décalage de une position à gauche - Multiplication par 10 (2).
II. 3 : 2 exercices d'union logique.
II. 4 : 2 exercices d'intersection logique.
II. 5 : 2 exercices de complémentation.
II. 6 : 1 exercice de complémentation incrémentée (de 1) (C'est une représentation d'un nombre opposé).
II. 7 : 2 exercices de somme d'un nombre et de son complément incrémenté (n'+ n).

. Chapitre III - Récapitulation en numération LN.

III. 1 : 1 exercice de conversion binaire → LN.
III. 2 : 1 exercice de conversion décimal → LN.
III. 3 : 1 exercice d'addition.
III. 4 : 1 exercice d'union.
III. 5 : 1 exercice d'intersection.
III. 6 : 1 exercice de complémentation.
III. 7 : 1 exercice de conversion LN → décimal.
III. 8 : 1 exercice de conversion LN → binaire.

N. B. : les nombres sur lesquels portent les exercices sont choisis aléatoirement.

Cette phase correspond aux § 1, 2 et 3 de l'analyse diachronique.

- Durée d'utilisation : ~ 30 mn.
- But de cette phase :
  . pratique immédiate et précise du langage binaire et du langage numérique LN utilisé.
  . pratique des principales opérations logiques et arithmétiques.
  . première approche du système conversationnel.

P 2 : PHASES EXER.

Il y a deux phases EXER (EXER 1 et EXER 2) qui sont conçues sur le même principe et qui assurent une progression dans l'apprentissage.

Cf. Figure II. 8.

. Le programme EXER comporte :
  - une simulation de la partie enseignée de l'ordinateur (" ordinateur-virtuel ").
  - un ensemble d'images " registres-mémoire " (exercices).
  - un programme de gestion (système).

. La communication élève ↔ système se fait par téléscripteur (TTY).

N. B. : l'ensemble d'exercices peut être changé à tout moment.

. Le système choisit (choix aléatoire) un exercice et le charge dans l'ordinateur-virtuel (image mémoire, image des registres).

. Le système indique à l'élève (impression) l'état de l'ordinateur-virtuel (image mémoire, image des registres, mode d'exécution).

. Le système active " l'ordinateur virtuel " pour le traitement d'une série d'étapes dans le " programme-exercice ", jusqu'à la rencontre d'une instruction de fin de programme : l'exercice est alors terminé.

. A chaque étape, le système :
  - demande à l'élève le contenu des registres.
  - enregistre les réponses (justes ou fausses), ne les refusant que si elles sont incohérentes (le nombre-réponse ne respecte pas le format du registre-question).
  - compare les nombres-réponses avec les valeurs réelles déterminées par l'ordinateur-virtuel.
  - s'il n'y a pas d'erreur, le système active l'ordinateur-virtuel pour passer à l'étape suivante.
  - s'il y a au moins une erreur, le système
    - mémorise le nombre d'erreurs.
    - et imprime un corrigé (image du pupitre) laissant à l'élève le soin de trouver lui-même ses erreurs, puis active l'ordinateur-virtuel pour passer à l'étape suivante.

→ deux exercices sont ainsi proposés.

P2 - 1 : phase EXER 1

Les exercices-programmes de la phase EXER 1 ne sont composés qu'avec des instructions à opérandes implicites.

Cette phase permet d'examiner un cas simple de fonctionnement de l'ordinateur (la mémoire n'est pas transformée).

Cette phase correspond aux § 4, 5, 6, et 7 de l'analyse diachronique.

P2 - 2 : phase EXER 2

Les " exercices-programmes " de la phase EXER 2 sont composés avec toutes les instructions à opérandes implicites ou explicites.

Cette phase permet d'examiner le fonctionnement complet de l'ordinateur (le contenu de la mémoire est utilisé et transformé). Seules les instructions de transfert ne sont pas utilisées.

Cette phase correspond aux § 8, 9, 10 et 11 de l'analyse diachronique.

- Durée d'utilisation :
  EXER 1 : $\sim$ 60 mn.
  EXER 2 : $\sim$ 90 mn.

- But de cette phase.
  EXER 1 est une préparation à EXER 2.

  . connaissance pratique précise du fonctionnement de l'unité de traitement de l'ordinateur sur les instructions autres que celles de transfert.
  . apprentissage rigoureux et rapide des instructions enseignées.
  . manipulations sur l'adressage.

P 3 : PHASE PROCES

Cf. Figure II. 9.

Cette phase est la prolongation logique des deux phases EXER précédentes.

. L'élève a cette fois l'accès direct à l'ordinateur-virtuel.

. " L'exercice " est cette fois un programme écrit par l'élève.

- Principe :

Le programme PROCES comporte :

- une simulation de la partie enseignée de l'ordinateur (" ordinateur-virtuel ").
  - une mémoire (zone disponible).
  - les registres
  - les interrupteurs de commandes et de contrôle.
- un programme de visualisations (demandées par commandes).
  - l'élève peut visualiser le pupitre complet.
  - l'élève peut ne visualiser qu'un seul registre.
- un programme de contrôle.
  - ce programme contrôle le travail effectué et signale les cas d'erreurs par messages :

ERR 1 : une adresse est hors de la zone disponible.
ERR 2 : le programme boucle (dépassement de temps).
ERR 3 : l'instruction courante est une instruction de transfert.

. Travail de l'élève :

- l'élève prépare un algorithme et le transcrit en langage machine.
- l'élève charge son programme en mémoire avec les interrupteurs de commandes.
- l'élève active son ordinateur avec ses interrupteurs de commandes.
- l'élève contrôle le contenu de la mémoire avec ces interrupteurs de contrôle.

N. B. : en cas d'erreur, l'élève doit commander l'exécution de son programme " pas à pas " et retrouve le travail effectué lors des phases EXER, mais sur un programme écrit, cette fois, par lui.

Cette phase correspond au § 12 de l'analyse diachronique.

- Durée d'utilisation : $\sim$ 120 mn.
- But de cette phase :
  - . renforcement des notions acquises avec les phases EXER.
  - . premier travail pratique de programmation (numérique).

P 4 : PHASE MONIT

Cf. Figure II. 10.

Cette phase est la prolongation logique de la phase PROCES précédente mais l'élève utilise un premier " software " élémentaire.

N. B. : arrivé à ce stade du cours, l'élève, non seulement utilise la phase MONIT, mais aussi en déchiffre la programmation. C'est-à-dire qu'il peut analyser les programmes (écrits dans le langage de l'ordinateur utilisé) : Moniteur, Editeur, Assembleur-Chargeur, Simulateur.

. Principe :

Le programme MONIT comporte :

1°) - UN MONITEUR :

. Il gère l'ensemble des travaux selon la demande (par commandes) de l'élève :

- appel à l'éditeur.
- appel à l'assembleur-chargeur.
- appel au simulateur.
- commande la fin du travail (annulation du fichier).

2°) - UN EDITEUR :

. Il permet l'édition d'une chaîne de lignes dans la " zone fichier ".

. L'élève dispose de 4 commandes :

- une commande d'insertion :
  . D'une (ou plusieurs) ligne (s) après une ligne donnée.
  . Pour commencer un fichier, ou pour insérer " avant " la première ligne d'un fichier existant, l'élève " insére " après une ligne fictive : " ligne 0 ".
- une commande de suppression :
  . D'une (ou plusieurs) ligne (s).
- une commande d'impression :
  . D'une (ou plusieurs) ligne (s).
- une commande de retour au moniteur.

3°) - UN ASSEMBLEUR-CHARGEUR.

. L'assembleur travaille en deux passages.
. Le premier passage crée une table d'identificateurs.
. Le deuxième passage et le chargeur travaillent conjointement.

3. 1. LE LANGAGE " SEMI-SYMBOLIQUE " UTILISE.

Pour une instruction, tout ce qui peut se représenter numériquement sans inconvénient (c'est le type de l'instruction, c'est aussi l'information complémentaire) se représente numériquement.

Une instruction à opérande implicite se représente entièrement numériquement.

Seule l'adresse de l'opérande, s'il y a lieu, se représente symboliquement (donc, commodités de corrections sous l'éditeur). Dans ce cas, la partie " adresse " de l'instruction est mise à zéro.

Lors de l'assemblage, il y aura addition au nombre " instruction " du nombre " opérande " (déterminé à partir de la table d'identificateurs).

3. 2. UTILISATION DE L'ASSEMBLEUR-CHARGEUR.

- L'élève peut demander (en option) l'impression de la liste d'assemblage :
  " adresse " " contenu " " ligne symbolique ".

4°) - UN SYSTEME D'EXECUTION (SIMULATEUR).

- En fait, le moniteur donne à l'élève l'accès à son ordinateur-virtuel (chargé).
- L'élève retrouve donc ce qu'il avait sous PROCES.
- L'élève dispose d'une commande supplémentaire pour le retour au moniteur.

. Travail de l'élève :

- L'élève analyse les programmes MONITEUR ; EDITEUR ; ASSEMBLEUR-CHARGEUR ; SIMULATEUR.
- L'élève prépare un algorithme et le transcrit en " semi-symbolique ".
- L'élève " édite " son programme.
- L'élève en commande l'assemblage (et le chargement).
- L'élève en dirige l'exécution comme il en a l'habitude.

Cette phase correspond aux § 18 et 19 de l'analyse diachronique.

- Durée d'utilisation : $\sim$ 120 mn.

- But de cette phase :

. Renforcement des connaissances acquises avec la phase PROCES.
. Analyse de programmes (moniteur, éditeur, assembleur, simulateur).
  - traitement de tables, de chaînes, des sous-programmes ...
. Révision complète du cours par l'analyse du programme-simulateur.
. Etude pratique analytique d'un software de base élémentaire réduit à l'essentiel.
  - cette étude analytique donne une connaissance globale de la nature et du rôle d'un software de base.

FIGURE II.8

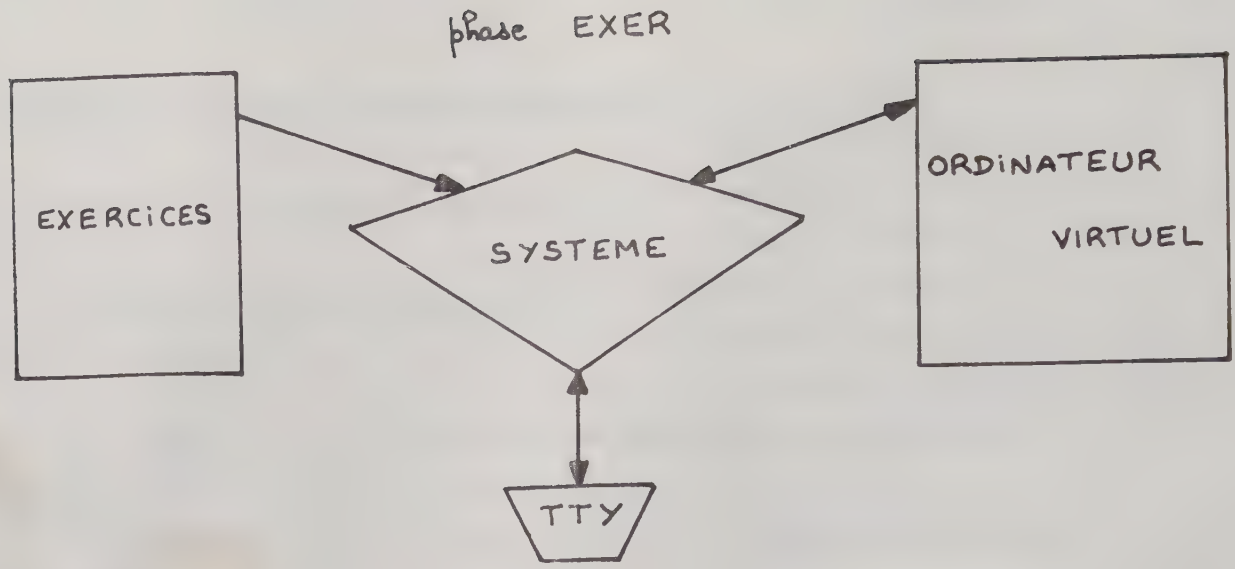


FIGURE II.9

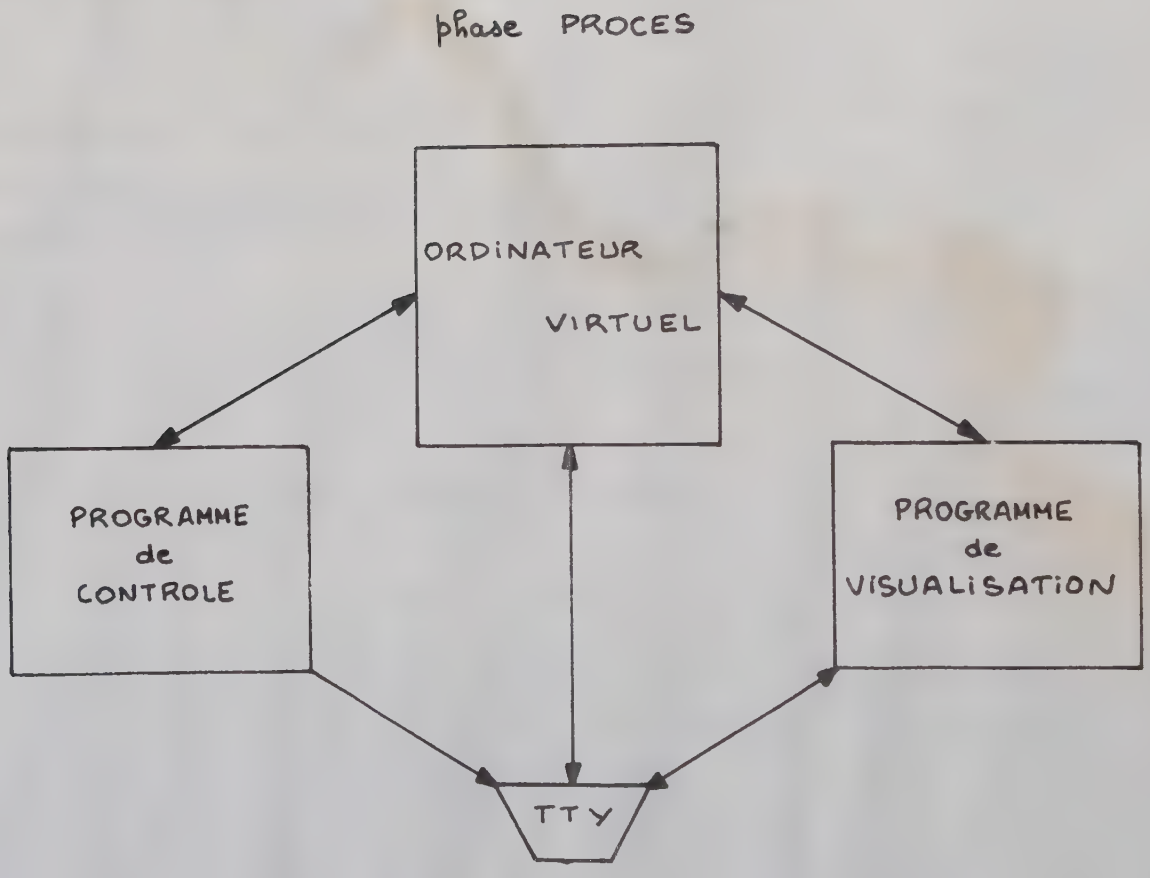
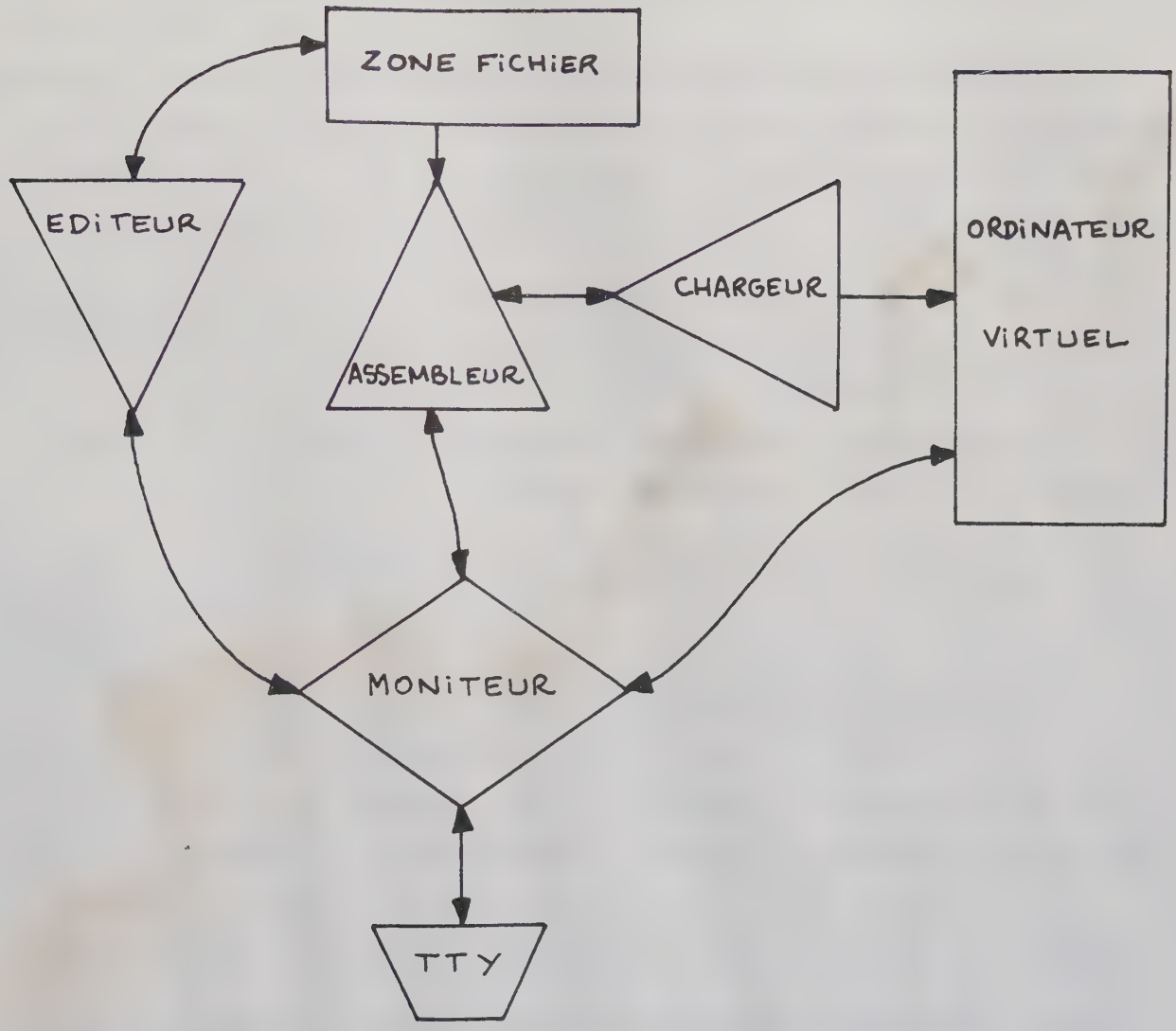


FIGURE II.10

phase MONIT



4°) - LE TEMPS DE SYNTHESE.

Ce quatrième temps est une conclusion (provisoire)..

L'élève a maintenant acquis un certain nombre de notions nouvelles (il a tout d'abord compris les motivations. Il a examiné ensuite, et analysé la réalité. Il a manipulé, enfin, ces nouvelles possibilités).

Avant d'aborder un nouveau cycle d'enseignement, traitant de nouveaux problèmes, il doit mettre en ordre ce qu'il vient d'acquérir. C'est-à-dire qu'il doit d'une part structurer, logiquement, son nouvel acquis ; d'autre part raccorder ce nouvel acquis aux connaissances antérieures.

En réalisant cet effort de coordination des connaissances, l'élève contrôle sa connaissance (donc la renforce).

Ce quatrième, et dernier, " enseignement ", doit être un enseignement individuel. C'est le rôle d'un manuel.

Quel manuel l'élève va-t-il utiliser ?

. Un manuel rédigé par le constructeur pour expliciter son ordinateur ?

C'est impossible pour l'instant.

- Ce manuel n'a pas été conçu pour des élèves débutants s'intéressant plus à la programmation en général, qu'à cet ordinateur en particulier, mais pour des informaticiens s'intéressant précisemment à cet ordinateur.

- Il est difficile, dans un tel manuel, d'isoler les notions connues des autres notions non encore étudiées (ou qui ne le seront jamais).

N. B. : Cependant, certains manuels publiés par les constructeurs, sont suffisamment clairs pour être cités comme documentation référentielle [F 1].

. Un cours " polycopié " rédigé de façon " traditionnelle " ?.

Cette solution est envisageable, mais pas parfaite.

- Un cours polycopié n'est généralement pas suffisant pour l'élève qui n'a pas suivi régulièrement le cours. Mais il n'est pas, réellement, fait pour lui. Et nous ne nous intéressons pas, dans ce texte, à de tels élèves.

- Le cours polycopié est un manuel malcommode pour l'élève qui suit régulièrement les cours.

. Un tel manuel n'a qu'un début et n'a qu'une fin : un seul cours y est donc traité : le cours complet.
Son emploi ne devient donc aisé qu'à la fin du cours.

. Pour une partie précise du cours traitée, le cours polycopié réexplique tout en détail : ces détails sont superflus si le cours a été bien effectué.

. En résumé : le cours polycopié, ainsi conçu, n'est pas un " outil de travail " approprié au cours réalisé. Il est une sorte d'outil parallèle, annexe (et, seulement pris dans ce sens, il n'est pas contestable).

N. B. : Là encore, certains de ces cours polycopiés pourront être cités comme documentation référentielle.

. Un manuel-outil conçu " pour " l'enseignement ?

C'est la solution adoptée.

. Conception de ce manuel (M 3).

- Ce manuel est divisé en 8 chapitres.

- Ces chapitres sont indépendants = chaque chapitre est, à lui seul, un petit manuel traitant la connaissance de l'instant = tout y est dit (les nouvelles connaissances sont intégrées aux anciennes).

Il n'y a rien de superflu.

- Mode de rédaction : le texte est aussi succint que possible. L'essentiel est dit par schémas (lecture facile, mais précise).

- L'assiduité aux trois temps pédagogiques précédents est nécessaire pour que l'utilisation de ce manuel soit pleinement profitable.

M 3 (1) : Premier chapitre.

. Schéma : structure globale d'un ordinateur.

. Texte : description succinte de l'unité mémoire.

. Schéma : unité mémoire.

. Schéma : principe de la sélection, de la lecture, de l'écriture. (cycle d'hystérésis, matrice de tores).

Ce chapitre correspond aux § 2 et 3 de l'analyse diachronique.

M 3 (2) : Deuxième chapitre.

. Texte : description succinte de l'unité de traitement (incluant le rôle de l'unité mémoire).

. Texte : description succinte du pupitre de contrôle et de commandes.

. Tableau : les interrupteurs de commandes.

. Schéma : structure globale de l'ordinateur.

Ce chapitre correspond aux § 4 et 5 de l'analyse diachronique.

M 3 (3) : Troisième chapitre.

. Tableau : instructions à opérandes implicites.

. Schéma : organigramme logique du traitement de ces instructions. (tout le cours y est résumé : rôle de l'unité mémoire, rôle de l'unité de traitement).

. Exemples : traitement de quelques programmes.

Ce chapitre correspond aux § 6 et 7 de l'analyse diachronique.

M 3 (4) : Quatrième chapitre.

. Texte : description succinte des problèmes de l'adressage (inventaire des conventions ; fonctionnement).

. Texte : description succinte des différents " états " du processeur.

. Schéma : organigramme global de fonctionnement de l'ordinateur.

Ce chapitre correspond aux § 8 et 9 de l'analyse diachronique.

M 3 (5) : Cinquième chapitre.

. Tableau : les instructions à opérandes explicites et rappel des instructions précédemment étudiées.

. Schéma : organigramme logique du traitement de ces instructions (mise au point précise sur l'ensemble du cours).

. Exemples: traitement de quelques programmes.

Ce chapitre correspond aux § 10 et 11 de l'analyse diachronique.

M 3 (6) : Sixième chapitre.

. Texte : description succinte d'un système d'entrées/sorties.

. Tableau : instructions pour le téléscripteur et interruptions. (+ récapitulation des autres instructions).

. Schéma : structure précise de l'ordinateur (unités de transferts et de commandes ; unité arithmétique et logique ; unité de traitement ; unité mémoire).

. Exemples: programmes utilisant les instructions de transfert.

Ce chapitre correspond aux § 13 et 14 de l'analyse diachronique.

M 3 (7) : Septième chapitre.

. Texte : description succinte des mémoires annexes magnétiques.

. Schéma : leur fonctionnement.

. Schéma : diagramme technologique du fonctionnement.

M 3 (8) : Huitième chapitre.

. Texte : description succinte d'un assembleur.

. Schéma : composantes et fonctionnement d'un assembleur.

. Texte : généralisations.

Ce chapitre correspond aux § 18 et 19 de l'analyse diachronique et s'intégre au § 20.

II § 3 - SCHEMAS SYNOPTIQUES -

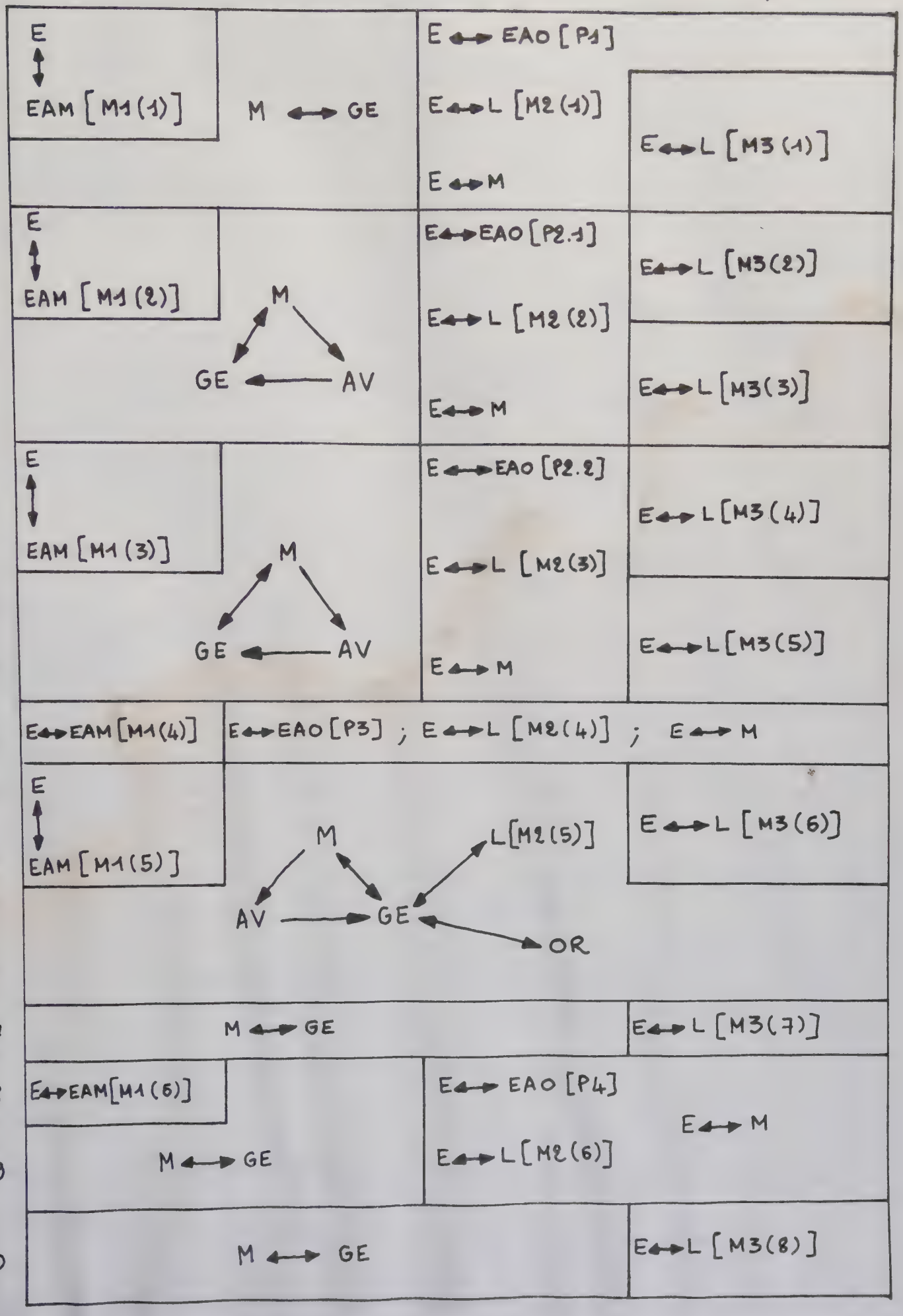
Pour clore cette description du système SCALP, je vous propose ces schémas synoptiques qui " visualisent " l'ensemble du système.

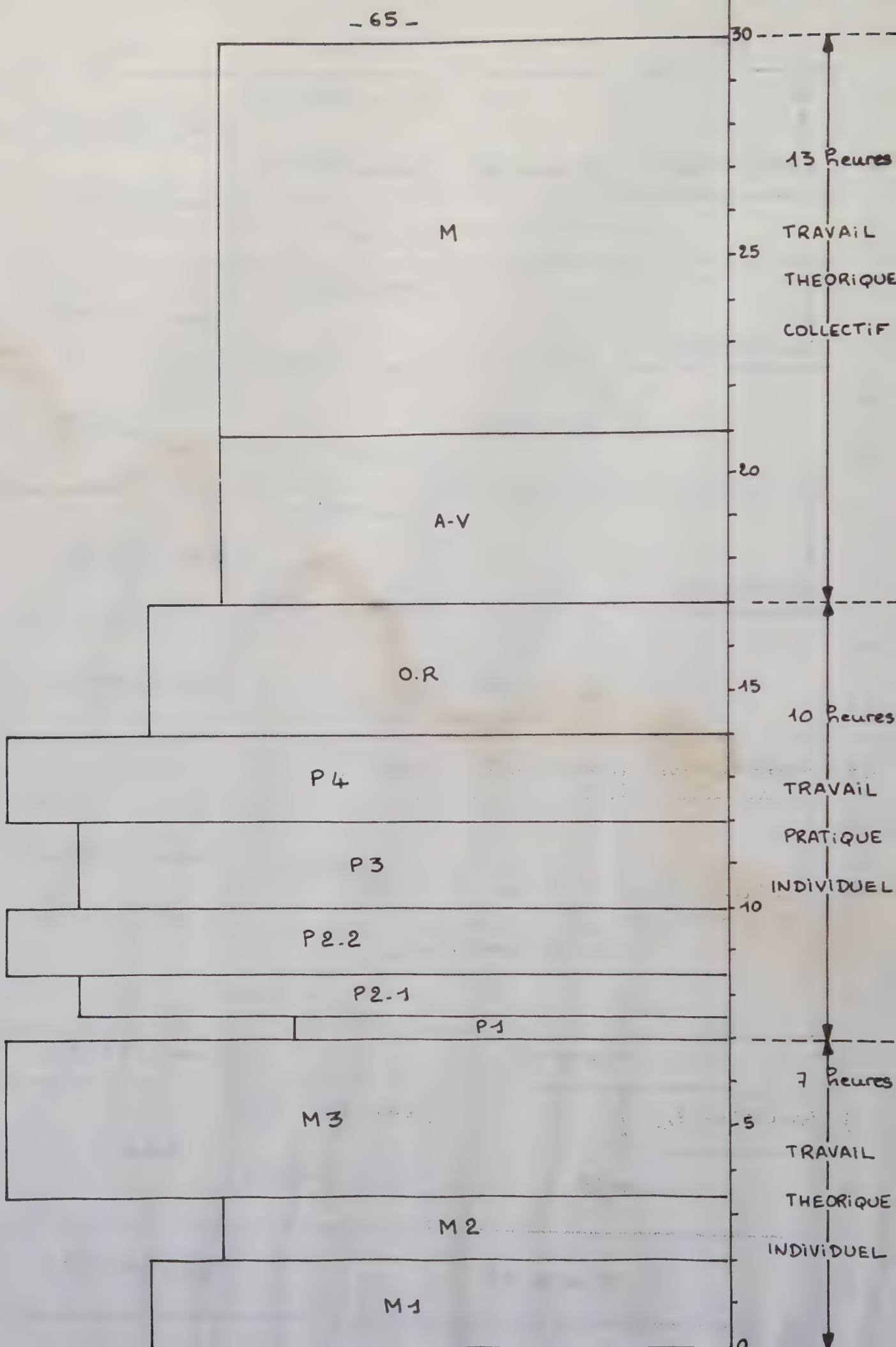
SCHEMA 1 : Verticalement, c'est l' axe diachronique, et ses 20 centres d'intérêt (1, 2, 3,...., 19, 20).
Horizontalement, c'est l'axe synchronique, et ses 4 temps pédagogiques (T1, T2, T3, T4).

SCHEMA 2 : Verticalement, c'est le temps d'utilisation de chaque " outil " d'enseignement.
Horizontalement, c'est l'importance-relative-de ces différents outils.

LEGENDE :

AV : ensemble audio-visuel.
E : élève.
EAM : enseignement assisté par manuel.
EAO : enseignement assisté par ordinateur.
GE : groupe d'élèves.
L : livre.
M : maître.
M 1 (i) : i^e chapitre de M 1.
M 2 (i) : i^e chapitre de M 2.
M 3 (i) : i^e chapitre de M 3.
OR : ordinateur réel.
P 1 : phase NUMER.
P 2. 1 : phase EXER 1.
P 2. 2 : phase EXER 2.
P 3 : phase PROCES.
P 4 : phase MONIT.


T1
T2
T3
T4
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
E
EAM [M1(1)]
M ⟷ GE
E ⟷ EAO [P1]
E ⟷ L [M2(1)]
E ⟷ M
E ⟷ L [M3(1)]
E
EAM [M1(2)]
M
GE
AV
E ⟷ EAO [P2.1]
E ⟷ L [M2(2)]
E ⟷ M
E ⟷ L [M3(2)]
E ⟷ L [M3(3)]
E
EAM [M1(3)]
M
GE
AV
E ⟷ EAO [P2.2]
E ⟷ L [M2(3)]
E ⟷ M
E ⟷ L [M3(4)]
E ⟷ L [M3(5)]
E ⟷ EAM [M1(4)]
E ⟷ EAO [P3] ; E ⟷ L [M2(4)] ; E ⟷ M
E
EAM [M1(5)]
M
L [M2(5)]
AV
GE
OR
E ⟷ L [M3(6)]
M ⟷ GE
E ⟷ L [M3(7)]
E ⟷ EAM [M1(6)]
M ⟷ GE
E ⟷ EAO [P4]
E ⟷ L [M2(6)]
E ⟷ M
M ⟷ GE
E ⟷ L [M3(8)]


30
25
20
15
10
5
0
M
A-V
O.R
P 4
P 3
P 2.2
P 2.1
P 1
M 3
M 2
M 1
13 Heures
TRAVAIL
THEORIQUE
COLLECTIF
10 Heures
TRAVAIL
PRATIQUE
INDIVIDUEL
7 Heures
TRAVAIL
THEORIQUE
INDIVIDUEL

II § 4 : EVALUATION DU COUT DE L'IMPLANTATION DU SYSTEME -

1°) - LE MATERIEL.

. un "petit" ordinateur.

- mémoire de 12 à 16 K "mots-mémoire".
- un téléscripteur "principal".
- une mémoire annexe (tambour ou disques) : 300000 caractères.
- un lecteur de bandes magnétiques
- un multiplexeur et quinze téléscripteurs.
- un software de base (éditeur, assembleur, chargeur).

. un ensemble audio-visuel.

- un magnétoscope.
- une caméra - un micro.
- deux téléviseurs.

2°) - LA PROGRAMMATION.

- le système conversationnel : 0,3 h/année.
- les phases : 0,3 h/année.

3°) - LA REDACTION DES MANUELS (M 1, M2, M3).

0, 2 h/année.

4°) - LA PREPARATION EFFECTIVE DE L'ENSEIGNEMENT.

- programme précis de l'enseignement et réalisation de l'audio-visuel. 0,2 h/année.

* Soit, au total, environ 1 h/année.

II § 5 - SUBSEQUENCE -

1°) - UN CONTROLE AUTOMATIQUE DES CONNAISSANCES -

Le premier contrôle des connaissances peut-être fait par le maître lors du temps d'acquisition (travail pratique) puisque les élèves travaillent individuellement. Il est, cependant, possible d'envisager un contrôle " automatique ".

. 3 contrôles peuvent ainsi être intégrés aux phases :

Contrôle 1 : intégré à la phase NUMER.

Ce premier contrôle porterait sur la dernière partie de la phase. C'est-à-dire sur la partie " récapitulation " en langage numérique LN.

Le système totaliserait les erreurs et, en fin de phase, imprimerait pour chaque type d'exercices effectué le nombre d'exercices faux.

Types 1 et 2 : conversions (binaire, décimal) → LN.
Types 3, 4, 5, et 6 : opérations.
Types 7 et 8 : conversions LN → (binaire, décimal).

Contrôle 2 : intégré à la phase EXER 2.

Ce deuxième contrôle porterait sur le dernier exercice de la phase. Comme précédemment, le système totaliserait le nombre d'erreurs et en imprimerait le nombre, par type, c'est-à-dire par " registre-question ". D'autre part le système imprimerait le nombre de groupes de questions (correspondant à une étape de l'exécution de " l'exercice-programme ").

Contrôle 3 : intégré à la phase MONIT.

- Le système totaliserait toutes les erreurs.
- Lorsque l'élève estimerait que son programme est définitivement mis au point, il demanderait, par commande, le bilan de son travail.
- Le système imprimerait alors :
  1°) le "programme-symbolique " enregistré en zone fichier.
  2°) le nombre d'erreurs par type d'erreurs.

- commande illégale sous le moniteur.
- commande illégale sous l'éditeur.
- double définition de symbole.
- erreur syntaxique.
- symbole non défini.
- commande illégale sous le simulateur.
- erreur d'adresse à l'exécution.
- dépassement de temps. (Le programme boucle).
- instruction illégale.

3°) un début de trace de l'exécution du programme.

2°) - LA DISPONIBILITE DE L'ORDINATEUR.-

Il est intéressant de programmer les phases NUMER, EXER, PROCES et MONIT en langage d'assemblage propre à l'ordinateur enseigné et de les faire fonctionner sur cet ordinateur.

- les programmes ainsi rédigés peuvent-être analysés à la fin du cours.

- D'autre part l'ordinateur est ainsi " réservé " à l'enseignement et les seules contraintes sont propres à cet enseignement.

Mais lorsqu'une phase fonctionne, l'ordinateur est indisponible.
C'est parfois regrettable.
Si l'ordinateur était disponible, à tout moment, il serait possible d'amener, à tout moment, un élève (ou quelques élèves), devant l'ordinateur-réel, pour qu'il(s) puisse(nt) vérifier un acquis ou acquérir, plus vite, une notion mal comprise.

Pour rendre à l'ordinateur sa disponibilité, il suffit de programmer les phases NUMER, EXER, PROCES, MONIT sur un autre ordinateur.
(N. B. Seule alors la " partie-ordinateur-enseigné " est à acquérir : mémoire $\sim$ 4 k, 1 téléscripteur).

INTERET :

. Lorsqu'un groupe d'élèves travaille sur une certaine phase (fonctionnant sur l'autre ordinateur), le maître peut isoler du groupe une petite équipe (environ trois élèves) pour accomplir un travail complémentaire, en parallèle, sur l'ordinateur-réel.

TRAVAIL A ACCOMPLIR :

. Ce n'est qu'un simple travail de programmation.

Mais, la phase MONIT au moins, devra être tout de même " écrite " dans le langage d'assemblage de l'ordinateur enseigné, puisque ce programme est aussi un programme à analyser en fin de cours.

PROBLEME ESSENTIEL :

Le problème essentiel n'est pas un problème de programmation, mais un problème de matériel.

- Si les phases fonctionnent sur un système d'exploitation conversationnel qui n'est pas utilisé que pour l'enseignement, les contraintes ne seront plus propres à l'enseignement ce qui peut nuire à la liberté des maîtres et des élèves.

CHAPITRE III

EXPERIMENTATION

- L'ordinateur
- Le support software
- Organigramme logique de l'ordinateur enseigné
- Les phases programmées
- Bilan

III § 1 - L'ORDINATEUR -

Le P.D.P.8 de la Digital Equipment Corporation [C3].

- La mémoire.

. 3 champs (0,1 et 2) de 4 k mots de 12 bits.
. Chaque champ est décomposé en 32 pages de 128 mots.

- Les registres :

. Les registres de l'unité mémoire :
  - registre " adresse " : MA : memory address register (12 bits).
  - registre " tampon " : MB : memory buffer register (12 bits).
. le compteur ordinal :
  - le program counter (12 bits) : PC.
. le registre instruction :
  - l'instruction register (3 bits) : IR.
. les registres modifiables par programme :
  - l'accumulateur (12 bits) : AC.
  - le link (1 bit) : L. (prolongation " à gauche " de l'accumulateur).
. un registre externe modifiable par interrupteurs :
  - le switch register (12 bits).
. pour les extensions mémoires (champs 1 et 2).
  - Instruction Buffer Register (3 bits) ; Instruction Field Register (3 bits).
  - Data Field Register (3 bits).
  - Save Field Register (6 bits).

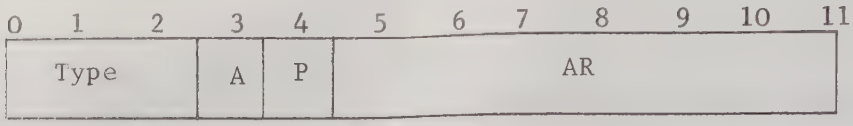
- Le générateur d'états : (Major State generator).

. pour les instructions :
  FETCH : lecture d'une instruction.
  DEFER : lecture d'une adresse " indirecte ".
  EXECUTE : lecture/écriture d'une valeur numérique à traiter/traitée.
. pour les " data break " :
  WORD COUNT, CURRENT ADDRESS, BREAK.

- Le cycle de base : 1,5 µs.

- Les instructions de longueur fixe : 1 mot (12 bits).

a) Opérande explicite :



bits 0 à 2 : type de l'instruction.

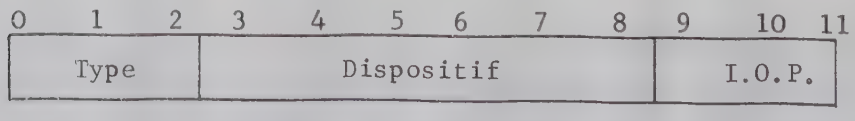
bit 3 : bit d'adressage : 0 : adressage direct.
1 : adressage indirect.

bit 4 : bit de page : 0 : page 0.
1 : page courante (celle de l'instruction).

bits 5 à 11 : adresse relative dans la page désignée par le bit 4.

- type 0 : AND Y.
  $[AC] \cap [Y] \rightarrow AC$.
- type 1 : TAD Y
  $[AC] + [Y] \rightarrow AC$.
- type 2 : ISZ Y
  1°) $[Y + 1] \rightarrow Y$; 2°) si $[Y] = 0$ alors $[PC] + 2 \rightarrow PC$ sinon $[PC] + 1 \rightarrow PC$.
- type 3 : DCA Y
  1°) $[AC] \rightarrow Y$; 2°) $0 \rightarrow AC$.
- type 4 : JMS Y
  1°) $[PC] + 1 \rightarrow Y$; 2°) $Y + 1 \rightarrow PC$.
- type 5 : JMP Y
  $Y \rightarrow PC$.

b) Transfert (type 6). IOT.



les I.O.P. (input-output pulses) commandent les actions.

c) Opérandes implicites : type 7 : OPR (micro-instructions).

2 groupes :

OPR1 :

0 1 2	3	4	5	6	7	8	9	10	11
type	0	CLA	CLL	CMA	CML	RL	RR	1/2	IAC

CLA : Clear accumulator. $0 \rightarrow AC$

CLL : Clear link. $0 \rightarrow L$

CMA : Complement accumulator $\overline{AC} \rightarrow AC$

CML : Complement link $\overline{L} \rightarrow L$

RL : rotate left
RR : rotate right
rotation (L-AC) à gauche/à droite.

1/2 : pour les rotations 1 ou 2 positions (0 ou 1).

IAC : increment accumulator $\overline{AC} + 1 \rightarrow AC$.

OPR2

0 1 2	3	4	5	6	7	8	9	10	11
type	1	CLA	SMA	SZA	SNL	RS	OSR	HLT	0

SMA : Skip on minus accumulator si $\overline{AC} < 0$ alors $\overline{PC} + 2 \rightarrow PC$
sinon $\overline{PC} + 1 \rightarrow PC$

SZA : Skip on zéro AC : si $\overline{AC} = 0$ alors $\overline{PC} + 2 \rightarrow PC$
sinon $\overline{PC} + 1 \rightarrow PC$

SNL : Skip on non zéro link : si $\overline{L} \neq 0$ alors $\overline{PC} + 2 \rightarrow PC$
sinon $\overline{PC} + 1 \rightarrow PC$

RS : reverse skips : si bit 8 = 1 les tests spécifiés aux bits 5, 6 et 7 sont inversés (SPA : $AC \geq 0$; SNA : $AC \neq 0$; SZL : $L = 0$).

OSR : Or with switch register : $\overline{AC} \cup \overline{SR} \rightarrow AC$

HLT : Halt : instruction d'arrêt.

- Les interrupteurs de commandes :

LA : Load address : $\overline{SR} \rightarrow PC$

DE : Deposit : $\overline{PC} \rightarrow MA$; $\overline{AC} \rightarrow MB$; écriture mémoire ; $\overline{PC} + 1 \rightarrow PC$

EX : Examine : $\overline{PC} \rightarrow MA$; lecture mémoire ; $\overline{PC} + 1 \rightarrow PC$

ST : Start : activation de l'ordinateur après remise à zéro des registres (sauf PC).

CO : Continue : activation de l'ordinateur sans remise à zéro.

STOP : Arrêt " manuel " de l'ordinateur.

SS : Single step : 2 positions : exécution normale ou cycle par cycle.

SI : Single instruction : 2 positions : exécution normale ou instruction par instruction.

. LES ORGANES PERIPHERIQUES :

- le télétype : communiquant avec l'ordinateur par des séries de 8 impulsions (les buffers d'entrée et de sortie ont donc 8 bits).
  . temps : 10 caractères/seconde.
- un multiplexeur : auquel sont connectés vingt télétypes.
- le tambour magnétique :
  . Composé de 4 champs de 64 K mots groupés en secteurs de 16 mots.
  - bit 256 K mots de 12 bits.
  . Temps moyen d'attente 10 ms.
- deux lecteurs de bandes magnétiques :
  . une bande contient 128 K mots de 12 bits.

N.B. : Ci-joint, le " flow-diagram " du P. D. P. 8 (cf Figure III. 1) décrivant :

- les états : FETCH, DEFER, EXECUTE.
- les états de " data break " : WORD COUNT, CURRENT ADDRESS, BREAK STATE.
- les opérations manuelles : LOAD ADRESS, START, CONTINUE, DEPOSIT, EXAMINE.
  . Ce flow diagram est utilisé dans le cours [A8].

FETCH

FiG. III. 1 / 1

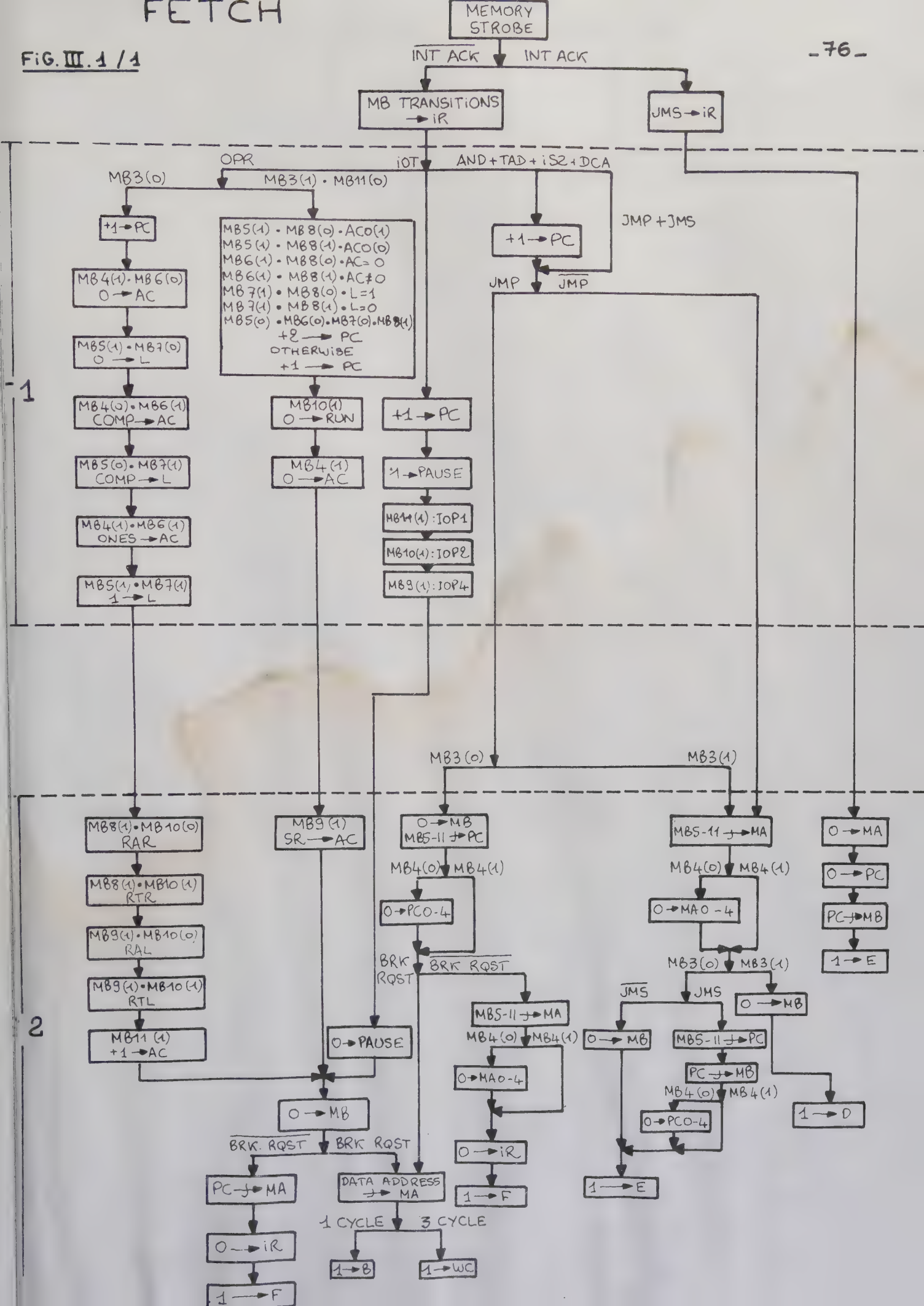


FiG. III.1/2

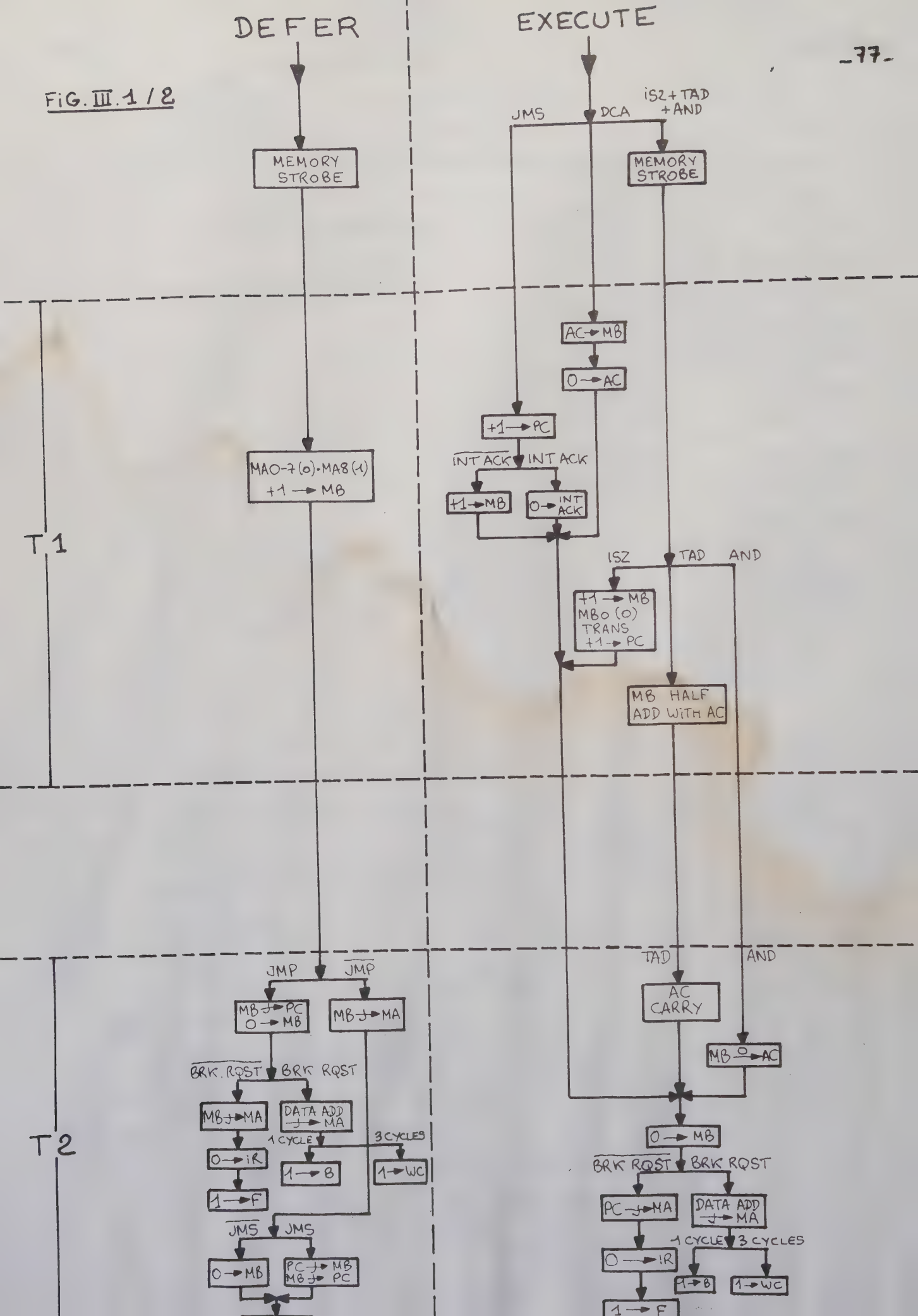
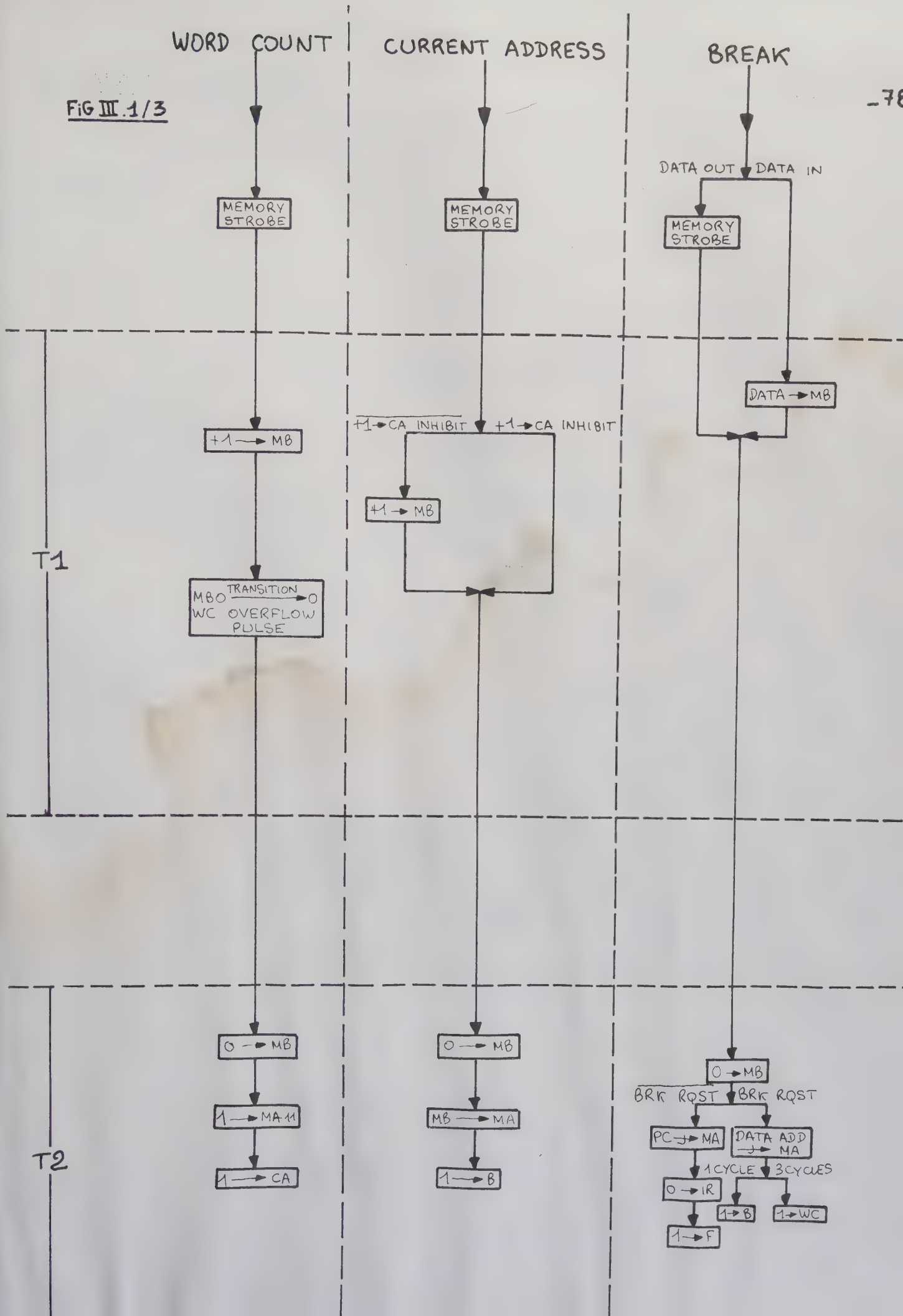
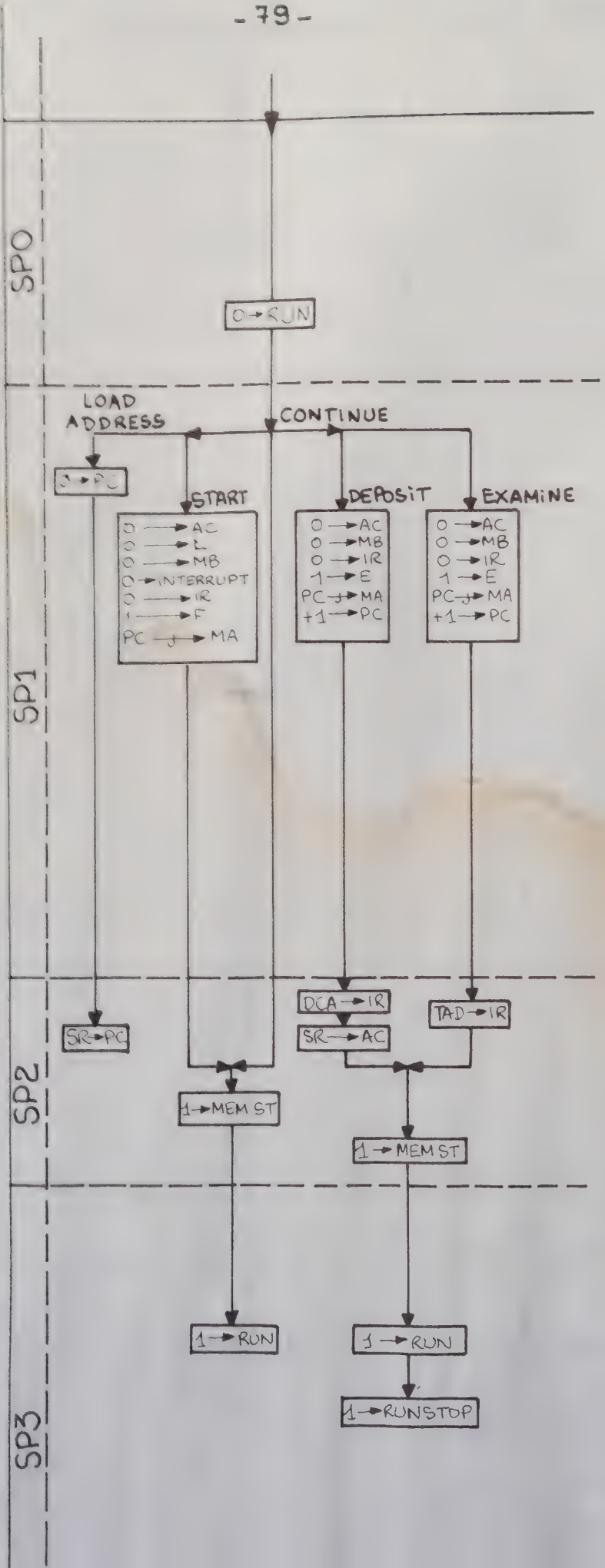

DEFER
EXECUTE
JMS
DCA
ISZ + TAD + AND
MEMORY STROBE
MEMORY STROBE
AC → MB
0 → AC
+1 → PC
INT ACK
INT ACK
+1 → MB
0 → INT ACK
MA0-7 (0) · MA8 (1)
+1 → MB
ISZ
TAD
AND
+1 → MB
MB0 (0)
TRANS
+1 → PC
MB HALF ADD WITH AC
T1
JMP
JMP
MB → PC
0 → MB
MB → MA
BRK RQST
BRK RQST
MB → MA
DATA ADD → MA
1 CYCLE
3 CYCLES
0 → IR
1 → B
1 → WC
1 → F
JMS
JMS
0 → MB
PC → MB
MB → PC
TAD
AND
AC CARRY
MB 0 → AC
0 → MB
BRK RQST
BRK RQST
PC → MA
DATA ADD → MA
1 CYCLE
3 CYCLES
0 → IR
1 → B
1 → WC
1 → F
T2

FiG III.1/3


WORD COUNT
CURRENT ADDRESS
BREAK
DATA OUT
DATA IN
MEMORY STROBE
MEMORY STROBE
MEMORY STROBE
DATA → MB
+1 → MB
+1 → CA INHIBIT
+1 → CA INHIBIT
+1 → MB
T1
MB0 TRANSITION → 0
WC OVERFLOW PULSE
0 → MB
0 → MB
0 → MB
BRK RQST
BRK RQST
1 → MA 11
MB → MA
PC → MA
DATA ADD → MA
T2
1 → CA
1 → B
1 CYCLE
3 CYCLES
0 → IR
1 → B
1 → WC
1 → F

COMMANDES MANUELLES

FIG III.1/4



III § 2 - LE SUPPORT SOFTWARE -

Le système SPARTACUS : système de programmation en ligne et en temps partagé.

Le Système SPARTACUS permet d'utiliser des programmes (phases) de 4 K mots en mode conversationnel et en temps partagé.
(N. B. les phases ont été programmées à l'aide du " Disk Monitor Systèm. D.E.C.") [C5]

Description succinte du système SPARTACUS :

- SPARTACUS fournit à chaque utilisateur un télétype, et une zone mémoire de 4 K (fichier individuel : F.I.) sauvegardé sur le tambour.

En fonctionnement :

- Le champ 0 contient le superviseur (SUPER).

Le superviseur gère le multiplexeur (donc les terminaux des utilisateurs) et gère les communications tambour ↔ champs 1 et 2.

- Pour les différents travaux, une " balance " est faite entre les champs 1 et 2.

Si un travail i (de l'utilisateur i) est en cours dans le champ 1 (ou 2), le superviseur recopie le champ 2 (ou 1) - travail (i-I) - dans la roue correspondante du tambour puis recopie la zone (i+I) du tambour dans le champ 2 (ou 1) : le travail (i+I) (de l'utilisateur (i+I) est prêt à fonctionner.

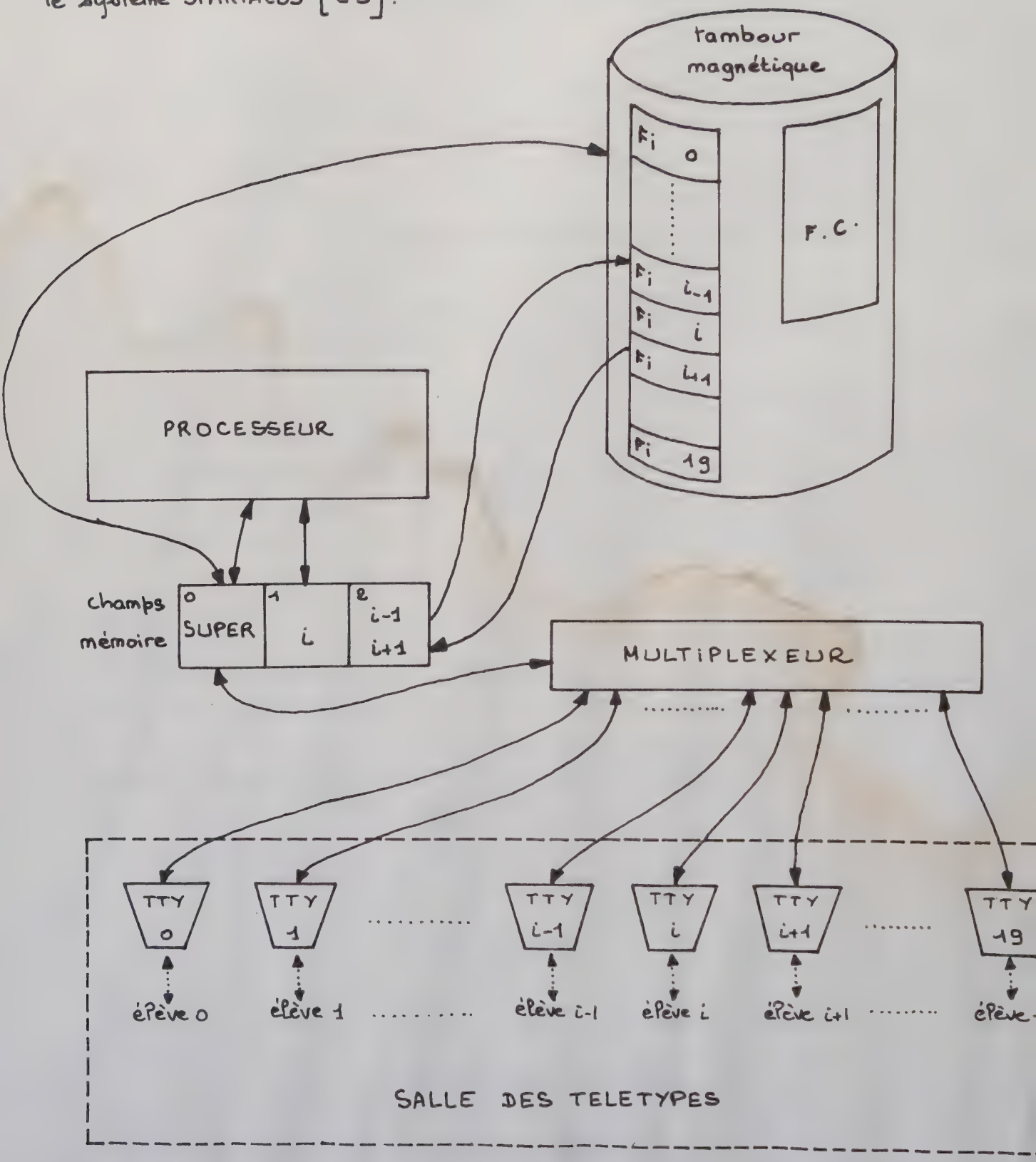
Le moniteur de SPARTACUS (INISYS) permet de charger dans une zone fichier-commun (F.C.), des phases-programmes de 4 K.

L'utilisateur, par commande ($ EXECU → " nom de la phase "), charge cette phase du fichier commun dans son fichier individuel.

Cf Figure III 2.

FIGURE III.2

le système SPARTACUS [C5].



III § 3 - ORGANIGRAMME LOGIQUE -

Cet organigramme est un extrait du manuel M3 - [A8] .

C'est le schéma donné dans le sixième chapitre - schéma décrivant le fonctionnement " logique " de l'ordinateur sur les instructions de types 0, 1, 2, 3, 4, 5, et 7.

Ce schéma est intéressant à plus d'un titre :

- Il est utilisé par l'élève à un moment central du cours. Ce moment correspond au passage entre la phase EXER 2 et la phase PROCES.

L'essentiel est " dit ", le reste n'est que conséquences.

- Il montre l'esprit réel de l'enseignement : l'étude est très précise (tout fait particulier est analysé) mais cette étude reste simple puisque seule la logique du fonctionnement est abordée (et non pas la technologie : à ce propos, il peut être intéressant de comparer ce diagramme logique avec le " flow-diagram " du P.D.P.8 donné dans le premier paragraphe de ce chapitre

- à noter que ce " flow-diagram " est lui même un schéma du manuel M3, mais ne venant que beaucoup plus tard, dans le dernier chapitre, à titre d'exemple).

Cf Figure III 3.

FIG. III.3

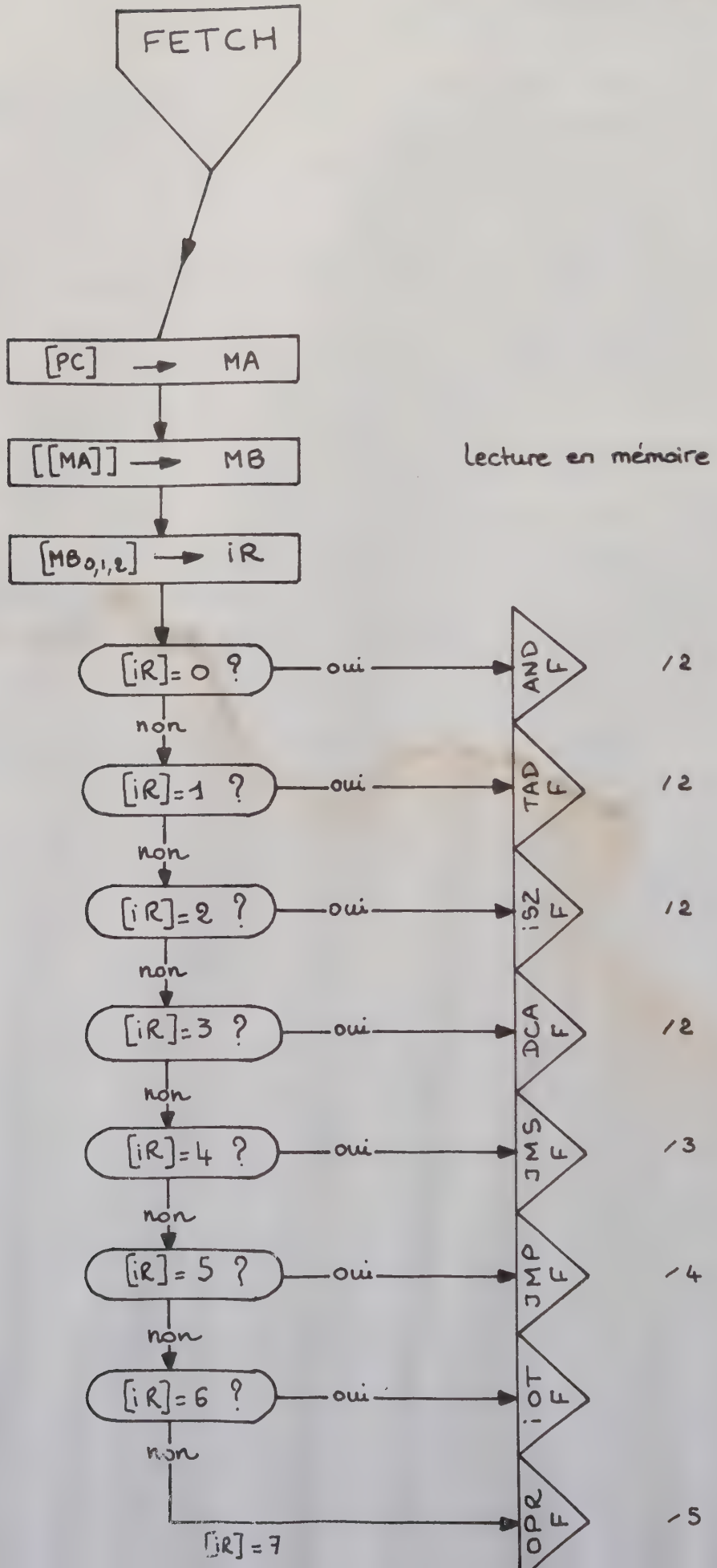

FETCH
[PC] → MA
[[MA]] → MB
lecture en mémoire
[MB0,1,2] → iR
[iR]=0 ?
oui
AND F
/2
non
[iR]=1 ?
oui
TAD F
/2
non
[iR]=2 ?
oui
iSZ F
/2
non
[iR]=3 ?
oui
DCA F
/2
non
[iR]=4 ?
oui
JMS F
/3
non
[iR]=5 ?
oui
JMP F
/4
non
[iR]=6 ?
oui
iOT F
non
OPR F
/5
[iR]=7

FIG. III.B/2

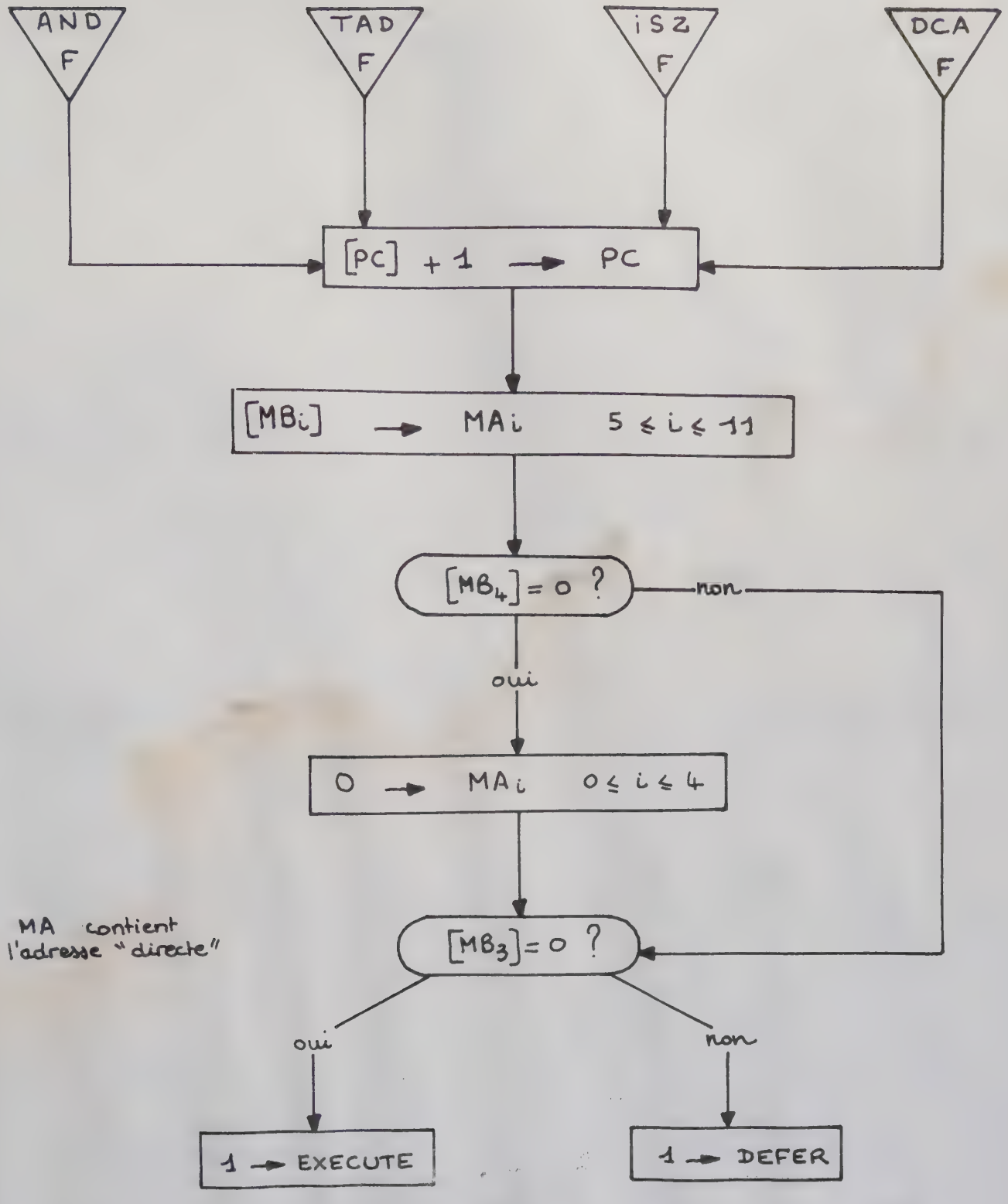

AND F
TAD F
iSZ F
DCA F
[PC] + 1 → PC
[MBi] → MAi 5 ≤ i ≤ 11
[MB4] = 0 ?
non
oui
0 → MAi 0 ≤ i ≤ 4
MA contient l'adresse "directe"
[MB3] = 0 ?
oui
non
1 → EXECUTE
1 → DEFER
/9
/8
adressage direct
adressage indirect

FIG. III.3

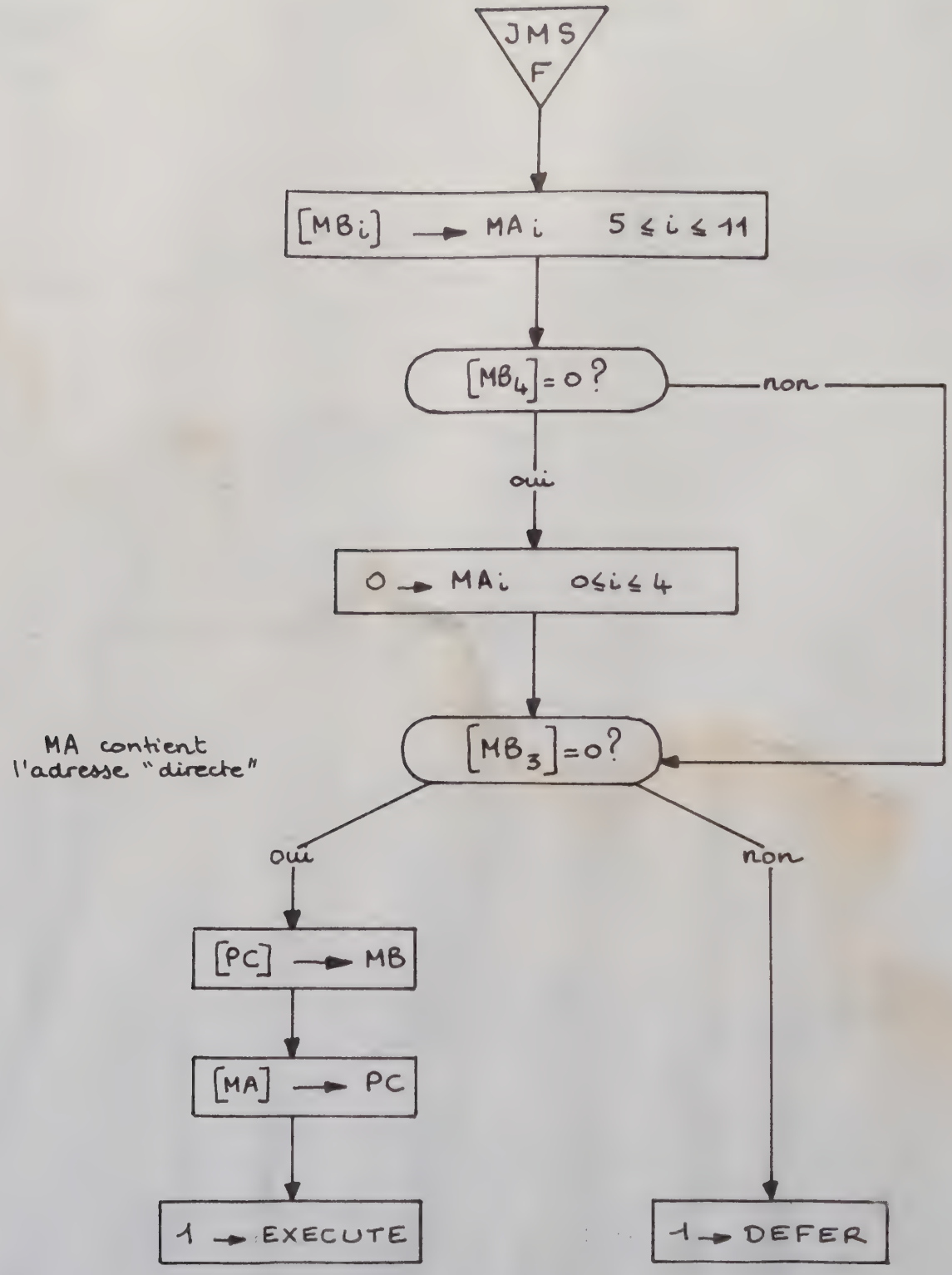

JMS F
[MBi] → MAi 5 ≤ i ≤ 11
[MB4] = 0 ?
non
oui
0 → MAi 0 ≤ i ≤ 4
MA contient l'adresse "directe"
[MB3] = 0 ?
oui
non
[PC] → MB
[MA] → PC
1 → EXECUTE
1 → DEFER
/9
/8
adressage direct
adressage indirect

FIG. III. 3 / 4

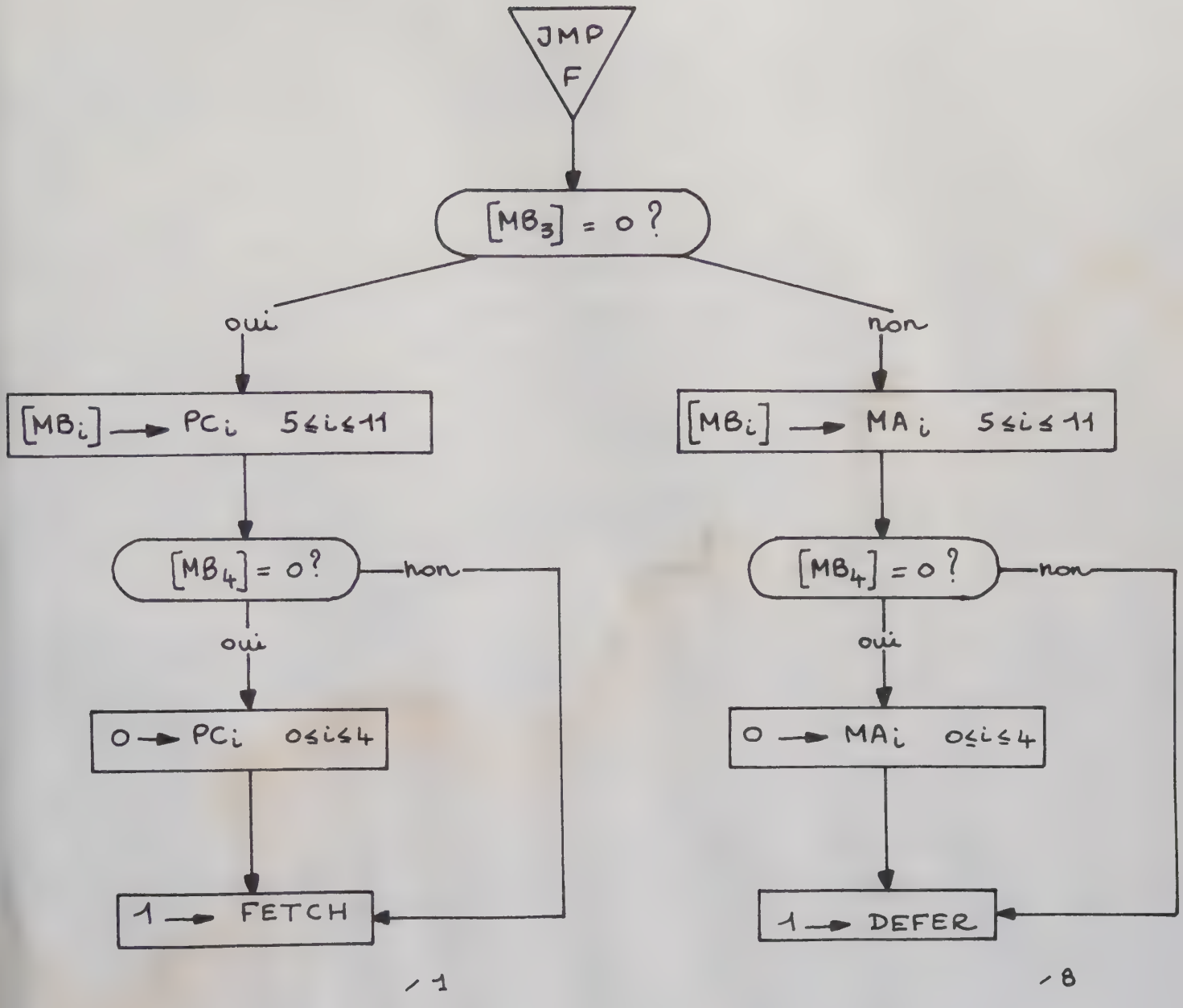

JMP F
[MB3] = 0 ?
oui
non
[MBi] → PCi 5≤i≤11
[MB4] = 0 ?
non
oui
0 → PCi 0≤i≤4
1 → FETCH
/ 1
adressage direct
PC contient l'adresse "directe"
[MBi] → MAi 5≤i≤11
[MB4] = 0 ?
non
oui
0 → MAi 0≤i≤4
1 → DEFER
/ 8
adressage indirect
MA contient l'adresse "directe"

FIG III.3 / 5

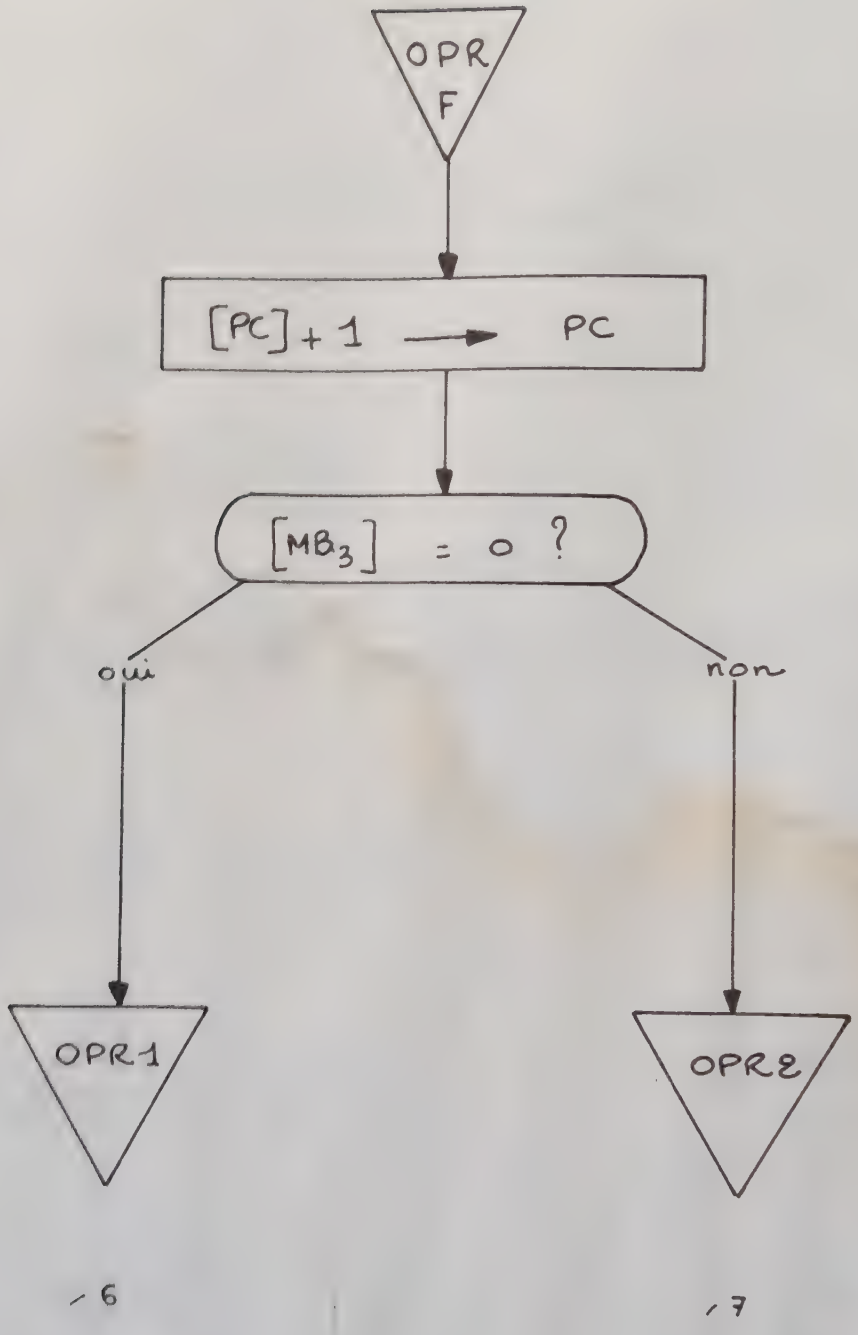


FIG III.3/6

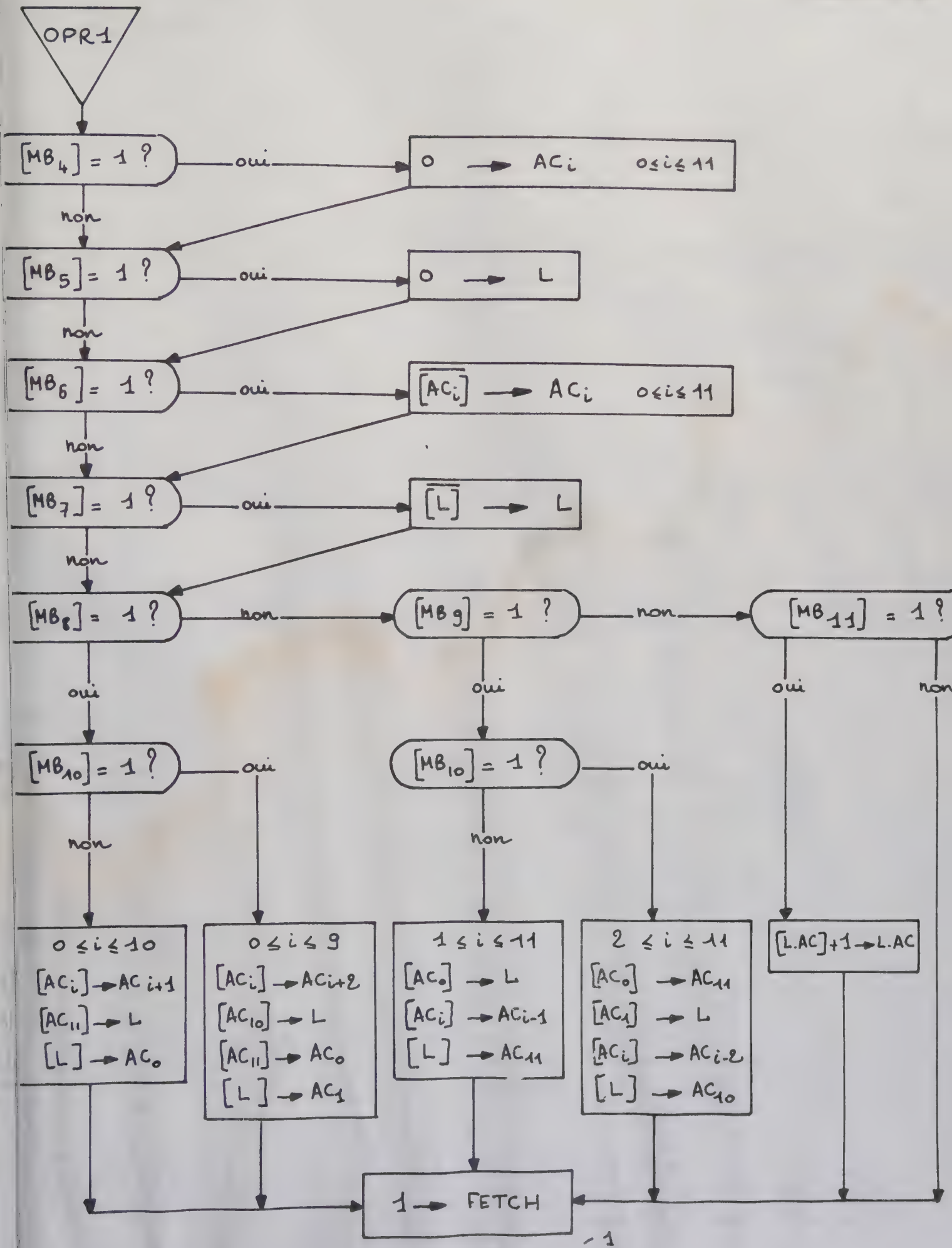

OPR1
[MB4] = 1 ?
oui
0 → ACi 0≤i≤11
non
[MB5] = 1 ?
oui
0 → L
non
[MB6] = 1 ?
oui
[ACi] → ACi 0≤i≤11
non
[MB7] = 1 ?
oui
[L] → L
non
[MB8] = 1 ?
non
[MB9] = 1 ?
non
[MB11] = 1 ?
oui
oui
oui
non
[MB10] = 1 ?
oui
[MB10] = 1 ?
oui
non
non
0 ≤ i ≤ 10
[ACi] → ACi+1
[AC11] → L
[L] → AC0
0 ≤ i ≤ 9
[ACi] → ACi+2
[AC10] → L
[AC11] → AC0
[L] → AC1
1 ≤ i ≤ 11
[AC0] → L
[ACi] → ACi-1
[L] → AC11
2 ≤ i ≤ 11
[AC0] → AC11
[AC1] → L
[ACi] → ACi-2
[L] → AC10
[L.AC]+1 → L.AC
1 → FETCH

FIG. III.5/7

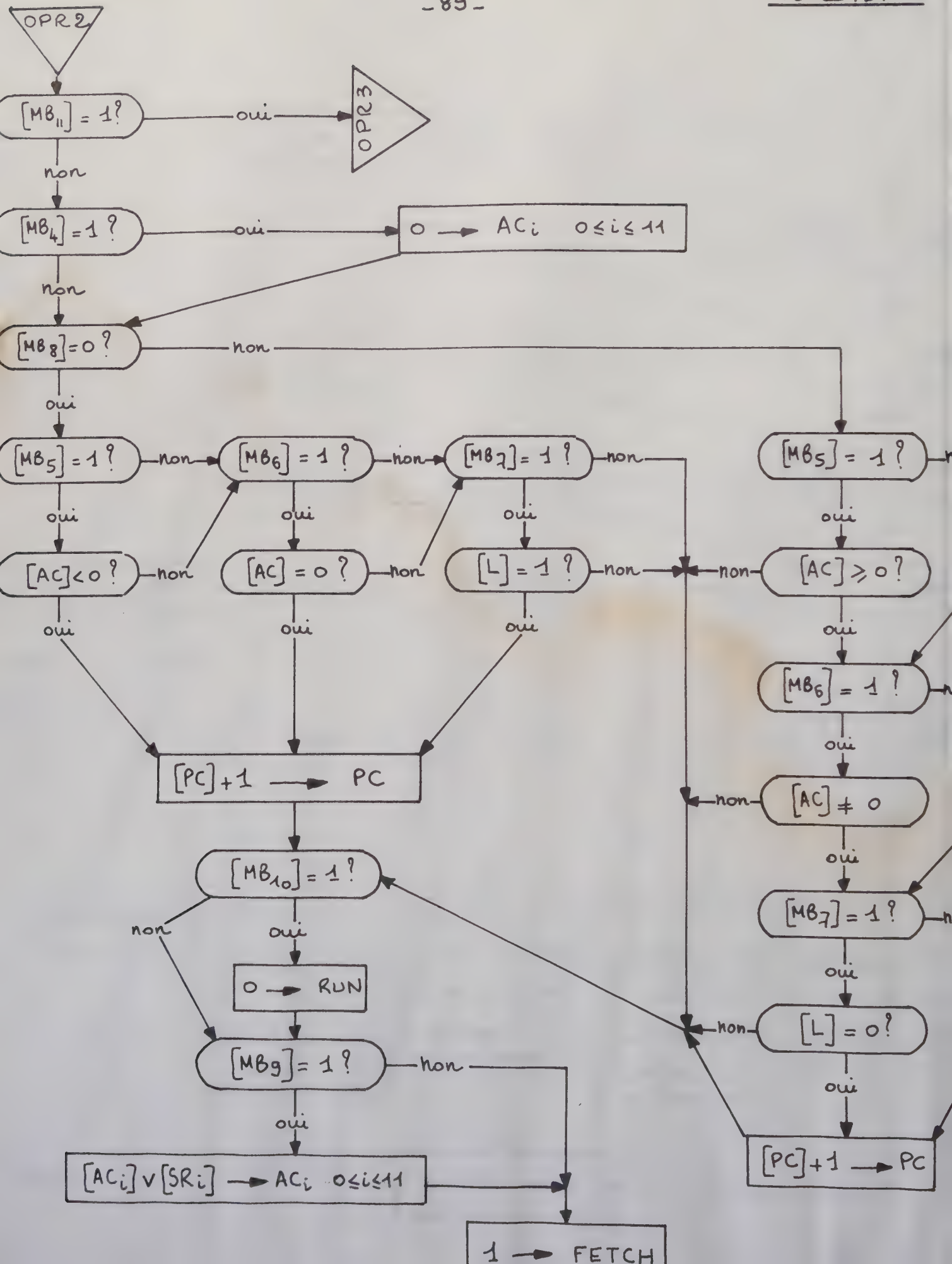

OPR2
[MB11] = 1 ?
oui
OPR3
non
[MB4] = 1 ?
oui
0 → ACi 0≤i≤11
non
[MB8] = 0 ?
non
oui
[MB5] = 1 ?
non
[MB6] = 1 ?
non
[MB7] = 1 ?
non
[MB5] = 1 ?
oui
oui
oui
oui
[AC] < 0 ?
non
[AC] = 0 ?
non
[L] = 1 ?
non
non
[AC] ≥ 0 ?
oui
oui
oui
oui
[MB6] = 1 ?
oui
[PC] + 1 → PC
non
[AC] ≠ 0
oui
[MB10] = 1 ?
oui
[MB7] = 1 ?
non
oui
oui
0 → RUN
non
[L] = 0 ?
[MB9] = 1 ?
non
oui
oui
[PC] + 1 → PC
[ACi] v [SRi] → ACi 0≤i≤11
1 → FETCH

FIG. III. 3 / 8

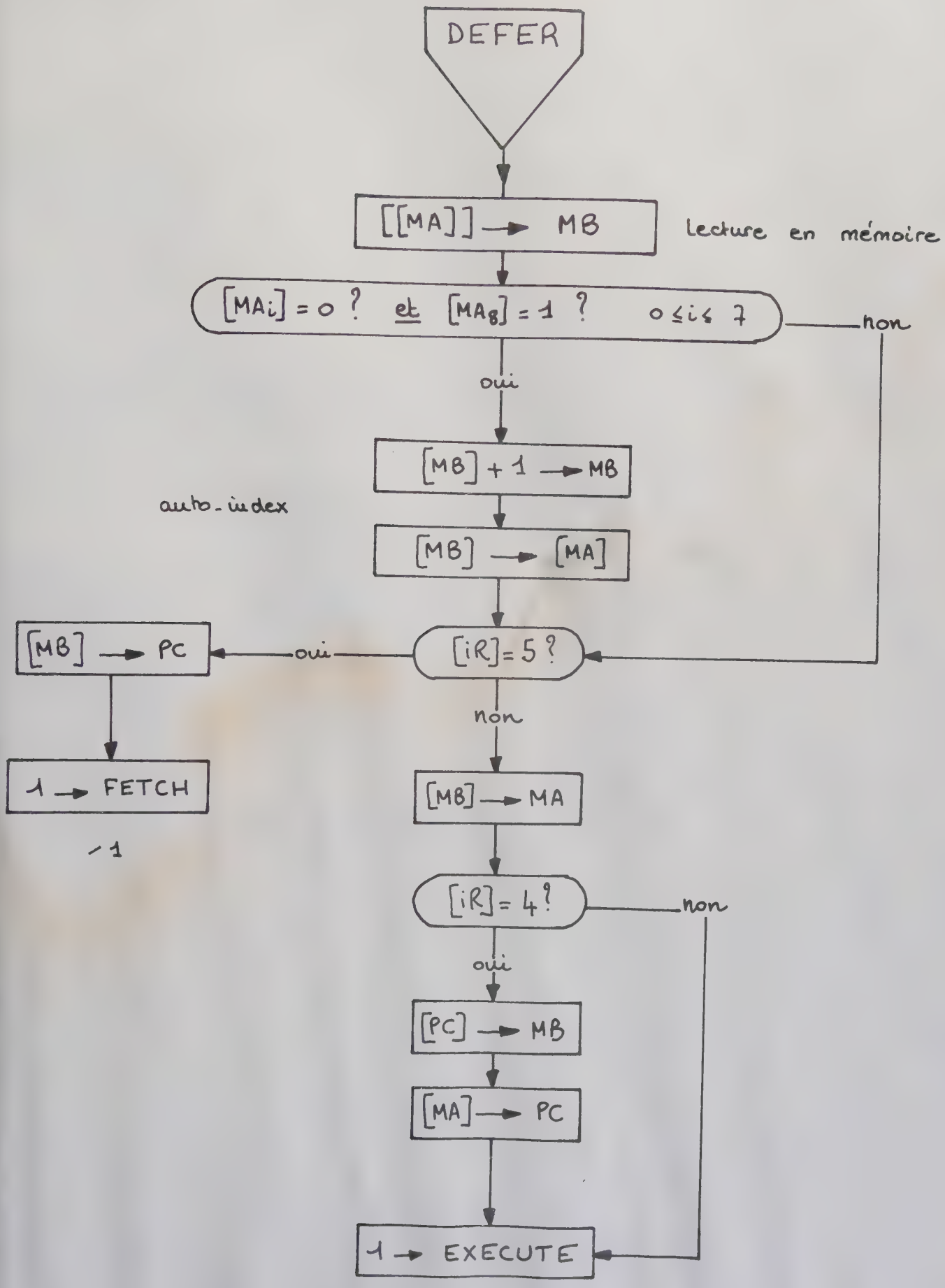

DEFER
[[MA]] → MB
lecture en mémoire
[MAi] = 0 ? et [MA8] = 1 ? 0 ≤ i ≤ 7
non
oui
[MB] + 1 → MB
auto-index
[MB] → [MA]
[MB] → PC
oui
[iR] = 5 ?
non
1 → FETCH
-1
[MB] → MA
[iR] = 4 ?
non
oui
[PC] → MB
[MA] → PC
1 → EXECUTE
-9

FIG. III.3/9

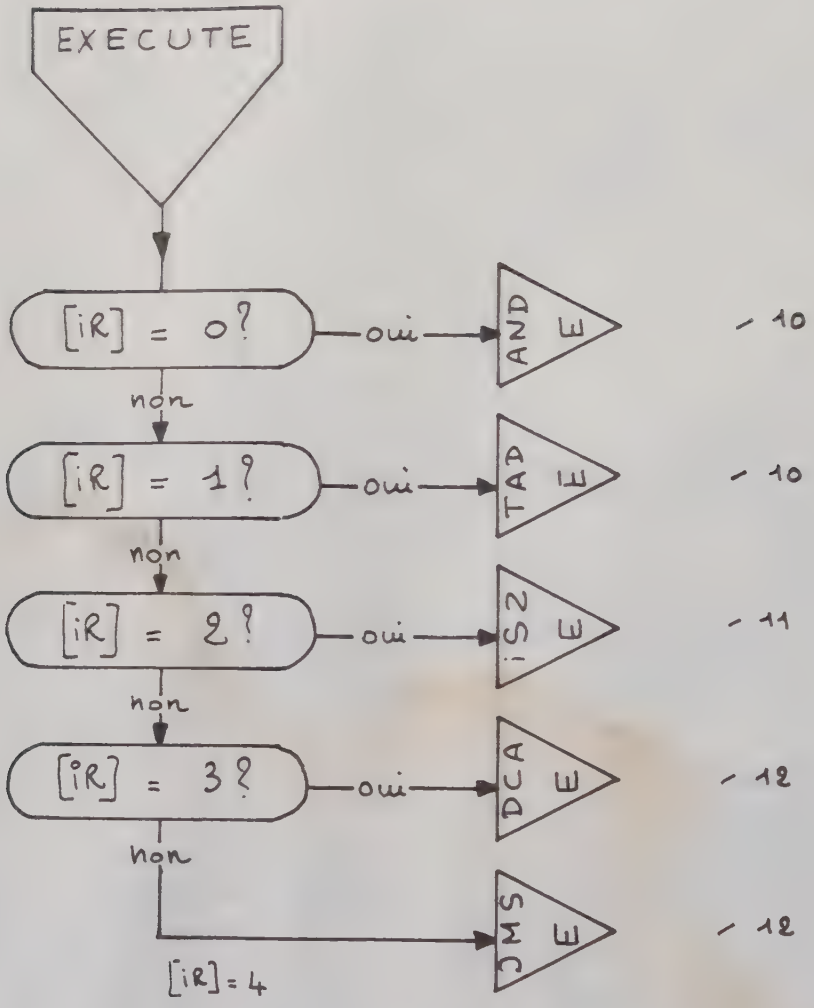

EXECUTE
[iR] = 0 ?
oui
AND E
- 10
non
[iR] = 1 ?
oui
TAD E
- 10
non
[iR] = 2 ?
oui
iSZ E
- 11
non
[iR] = 3 ?
oui
DCA E
- 12
non
[iR] = 4
JMS E
- 12

FIG. III . 3 / 10

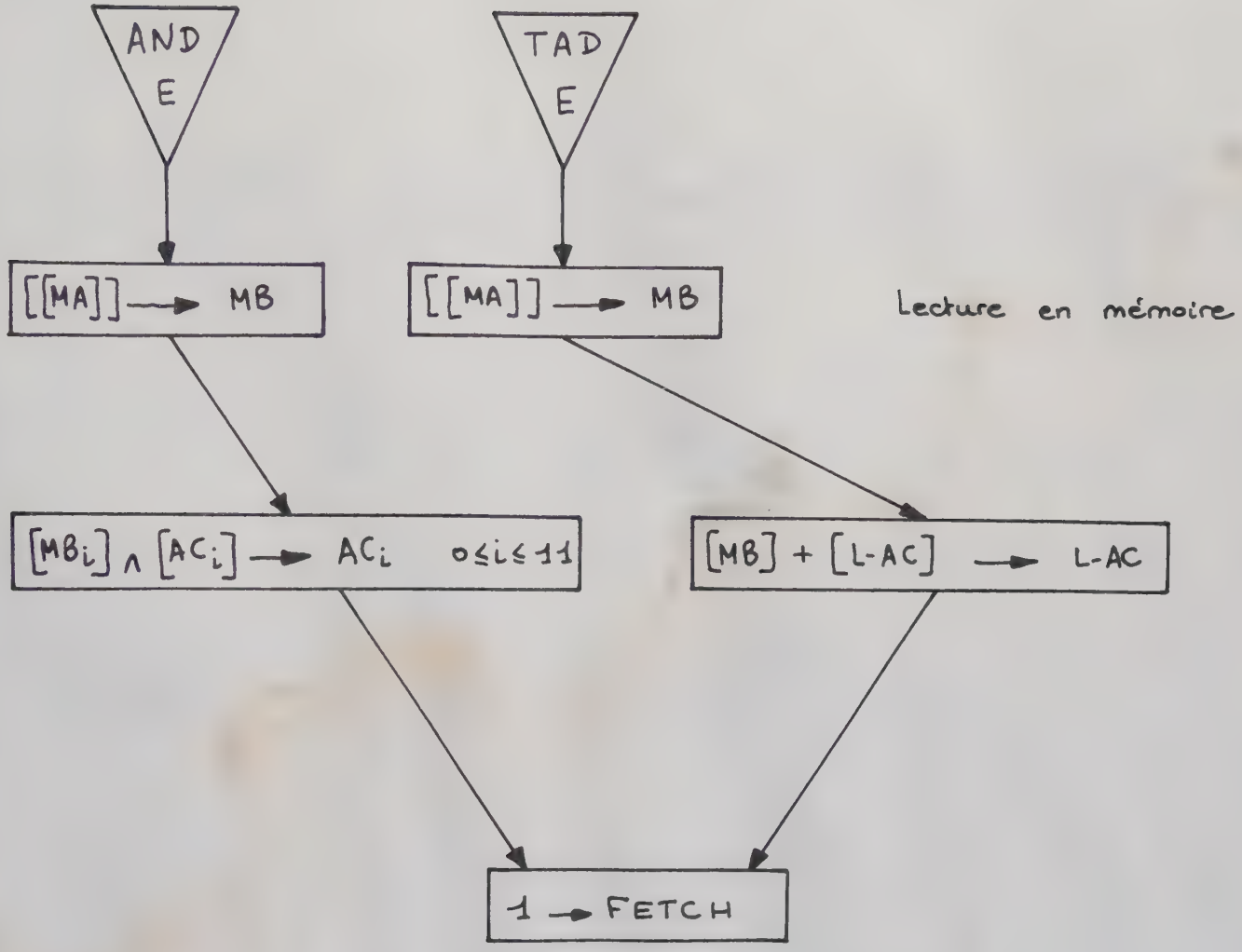

AND
E
TAD
E
[[MA]] → MB
[[MA]] → MB
Lecture en mémoire
[MBi] ∧ [ACi] → ACi 0≤i≤11
[MB] + [L-AC] → L-AC
1 → FETCH

FIG. III. 5 / 11

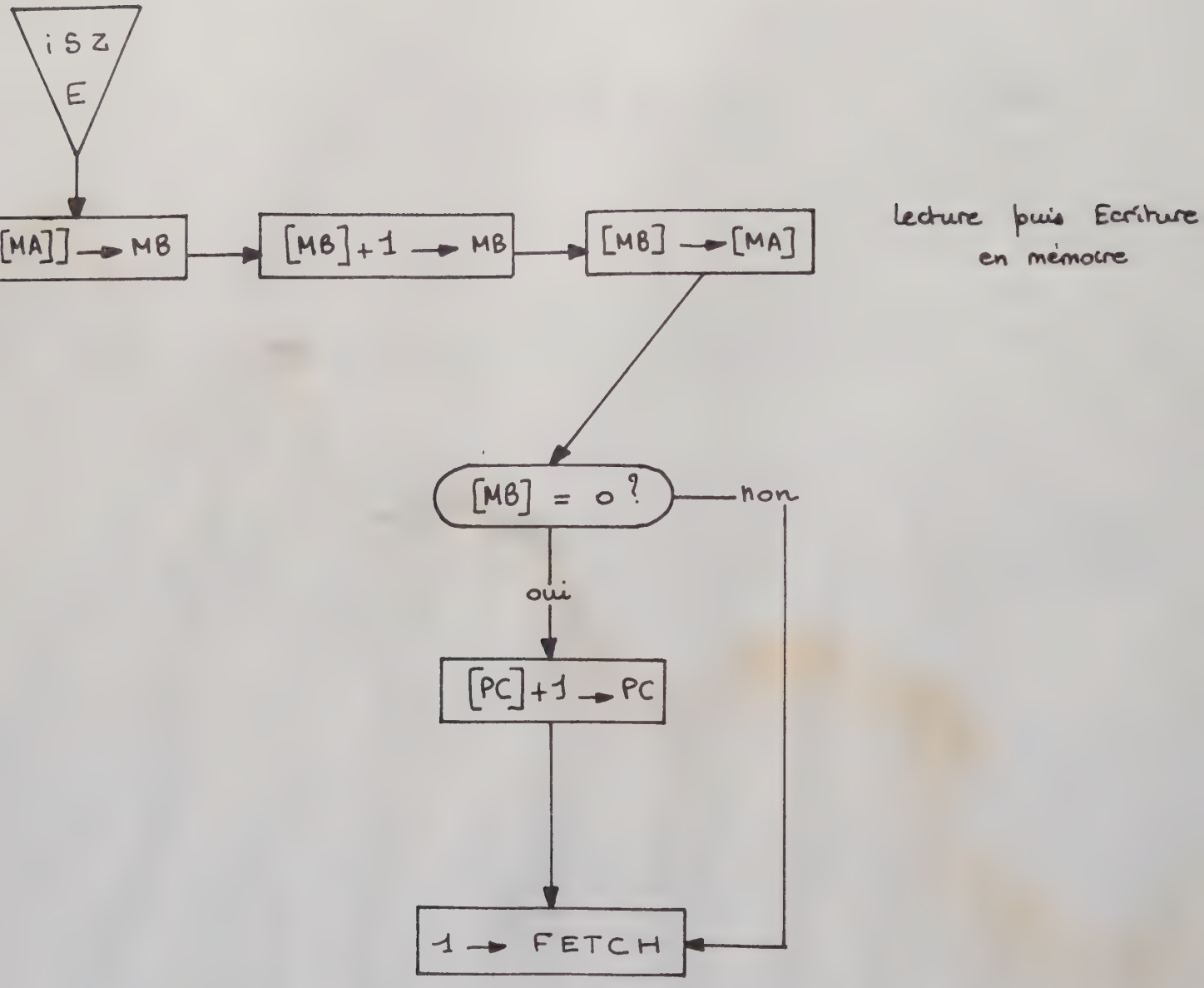
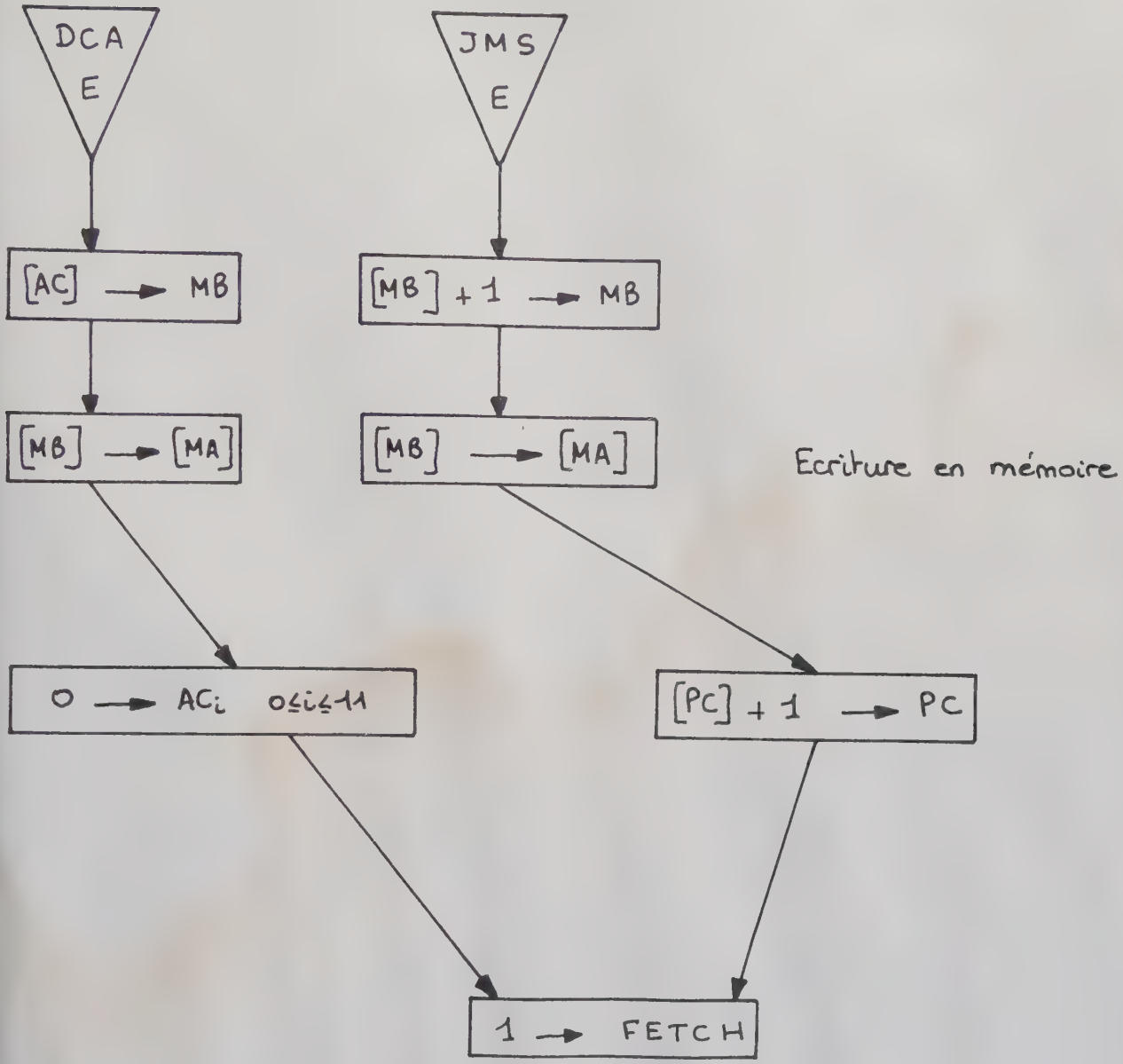

ISZ
E
[MA]] → MB
[MB]+1 → MB
[MB] → [MA]
lecture puis Ecriture en mémoire
[MB] = 0 ?
non
oui
[PC]+1 → PC
1 → FETCH

FIG. III. 3 / 12


DCA
E
JMS
E
[AC] → MB
[MB] + 1 → MB
[MB] → [MA]
[MB] → [MA]
Ecriture en mémoire
0 → ACi 0≤i≤11
[PC] + 1 → PC
1 → FETCH

III § 4 - LES PHASES -

1°) - LA PHASE NUMER -

. le langage numérique LN utilisé est le langage octal.

2°) - LES PHASES EXER -

. Les questions sont :
- l'état du processeur (fetch, Defer ou Execute).
- les registres : - instruction register.
  - program counter.
  - memory address.
  - memory buffer.
  - link.
  - accumulator.

3°) - LA PHASE PROCES -

Les requêtes sont :

SR, SS et SI pour charger le switch register, positionner les clefs de mode d'exécution.

ST, CO pour activer l'ordinateur.

PC, IR, MA, MB, L, AC pour visualiser les registres correspondants.

PU pour visualiser le pupitre dans son ensemble.

Les messages d'erreurs sont :

IOT??, BCLE??, ou ADR??

pour des erreurs-programmes.

?? pour une commande non comprise.

4°) - LA PHASE MONIT -

- requêtes sous le moniteur :
  - ED : appel de l'éditeur.
  - AS : appel de l'assembleur-chargeur.
  - EX : appel de l'exécuteur.
  - * * : fin de travail ; annulation du fichier.

- requêtes sous l'éditeur :

I : insertion, après une ligne donnée, d'une ou plusieurs lignes.

S : suppression d'une ou plusieurs lignes.

L : liste d'une ou plusieurs lignes.

F : retour au moniteur.

N. B. : les lignes possèdent 16 caractères au plus.

- requêtes sous l'assembleur :

une option permet de demander la liste d'assemblage.

LANGAGE SEMI-SYMBOLIQUE

< programme > :: = < Ligne > | < Ligne > < programme >

< Ligne > :: = < pseudo-instruction > | < instruction > | < valeur >

< pseudo-instruction > :: = ★ ␣ < adresse de chargement >

< adresse de chargement > :: = < nombre >

< instruction > :: = < instruction à opérande explicite > |
< instruction à opérandes implicites >

< instruction à opérandes implicites > :: = < étiquette > ␣␣ < représentation numérique >

< représentation numérique > :: = < nombre >

< étiquette > :: = ␣␣␣␣ | < symbole >

< instruction à opérande explicite > :: = < étiquette > ␣␣ < opérateur > ␣
< opérande >

< opérateur > :: = < nombre >

< opérande > :: = < symbole >

< valeur > :: = < étiquette > ␣␣ < nombre >

< nombre > :: = < chiffre octal > < chiffre octal > < chiffre octal > < chiffre octal

< chiffre octal > :: = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7

< symbole > :: = < lettre > < caractère > < caractère > < caractère >

< lettre > :: = A | B | C | D | E | F | G | H | I | J | K | L | M | N | O
P | Q | R | S | T | U | V | W | X | Y | Z

< caractère > :: = < lettre > | < chiffre octal > | ␣

```
$***
$EXECU  EXER
```

appel du superviseur

appel de la phase

EXERCICE 1

ETAT INITIAL DE LA MEMOIRE :

ADRESSE	CONTENU
0372	7012
0373	7110
0374	7420
0375	5772
0376	2372
0377	5772
0400	1201
0401	0200
0402	7010
0403	0207
0404	4607
0405	7010
0406	7402
0407	0372

ETAT INITIAL DU PUPITRE :

```
        PC 0000     AND    EXECUTE
        MA 0000
        MB 0000
   L 0  AC 0000
        SR 0400      SS 1    SI 0
```

EXEMPLE 1

phase EXER 2

données fournies par le système.

LOAD ADDRESS

START

} activation de l'ordinateur virtuel par le système

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :F

CONTENU DES REGISTRES :

CONTINUE

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :F

CONTENU DES REGISTRES :

IR 7
PC 0375
MA 0374
MB 7420
L 0
AC 0040

CONTINUE

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :F

CONTENU DES REGISTRES :

IR 5
PC 0375
MA 0375
MB 5772
L 0
AC 0040

Le système active l'ordinateur virtuel pour le passage à "l'étape" suivante.

CONTINUE

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :D

CONTENU DES REGISTRES :

IR 5
PC 0375
MA 0372
MB 7012
L 0
AC 0040

questions posées par le système

réponses fournies par l'élève

1 ERREUR

nombre d'erreurs

PC 0375 JMP DEFER
MA 0372
MB 0405
L 0 AC 0040
SR 0400 SS 1 SI 0

corrigé

CONTINUE

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :E

CONTENU DES REGISTRES :

IR 5
PC 7012
MA 0372
MB 0405
L 0
AC 0040

6 ERREURS

PC 0406 OPR FETCH
MA 0405
MB 7010
L 0 AC 0020
SR 0400 SS 1 SI 0

CONTINUE

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :F

CONTENU DES REGISTRES :

IR 7
PC 0407
MA 0406
MB 7402 ← instruction "7402" : "halt"
L 0
AC 0020 ← l'exercice est donc terminé.

EXERCICE TERMINE

EXERCICE 2

ETAT INITIAL DE LA MEMOIRE :

ADRESSE CONTENU

0512 0513
0513 7402
0514 2723
0515 5317
0516 5712
0517 1312
0520 0313

```
0521    5314
0522    7776
0523    0522

ETAT INITIAL DU PUPITRE :

          PC 0407    OPR   FETCH
          MA 0406
          MB 7402
     L 0  AC 0020
          SR 0514     SS 1   SI 0


LOAD ADDRESS

START

          QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :F


CONTENU DES REGISTRES :

IR 2
PC 0515
MA 0514
MB 2723
 L 0
AC 0020

1 ERREUR


          PC 0515    ISZ   FETCH
          MA 0514
          MB 2723
     L 0  AC 0000
          SR 0514     SS 1   SI 0


CONTINUE

          QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :D


CONTENU DES REGISTRES :

IR 2
PC 0515
MA 0523
MB 522
```

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :F

CONTENU DES REGISTRES :

IR 5
PC 0516
MA 0516
MB 5712
L 0
AC 000

BEFF

0402 ← l'élève annule sa réponse et en frappe une autre.

CONTINUE

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :D

CONTENU DES REGISTRES :

IR 5
PC 0516
MA 0512
MB 0513
L 0
AC 0402

CONTINUE

QUESTIONS :

ETAT DU PROCESSEUR (F,D,E) :F

CONTENU DES REGISTRES :

IR 7
PC 0514
MA 0513
MB 7402 ← le deuxième exercice est terminé.
L 0
AC 0402

EXERCICE TERMINE

TRAVAIL TERMINE ← fin de la phase

```
****
```
← appel du superviseur.

```
$EXECU  PROCES
```
← appel de la phase.

```
        MEMOIRE DISPONIBLE DE 0000 A 4577
```
} le système indique la z… mémoire utilisable par l'él…

EXEMPLE 2

phase PROCES

```
SR
SWITCH REGISTER 6000

DE
DEPOSIT

SR
SWITCH REGISTER 0000

LA
LOAD ADDRESS

ST
START
IOT??
```
← le simulateur a rencontré une instruction d'entrée / sortie (type 5)

```
PC
0000

SR
SWITCH REGISTER 5200

DE
DEPOSIT
```
} chargement du nombre 5200 (instruction : saut en 0000) à l'adresse 0000.

```
SR
SWITCH REGISTER 0000

LA
LOAD ADDRESS

ST
START
```
} exécution à partir de 0000

```
BCLE??
```
← le programme boucle

```
PC
0000

SR
SWITCH REGISTER 5601

DE
DEPOSIT

SR
SWITCH REGISTER 6784

DE
DEPOSIT

SR
SWITCH REGISTER 0000

LA
```

```
ADR??
```

le simulateur a rencontré une adresse hors de la zone mémoire disponible.

```
SR
SWITCH REGISTER 123
??
```

123 (en octal) représente 001010011 soit 9 bits. Le format n'est pas cohérent (SR : 12 bits).

```
PC
6704

SR
SWITCH REGISTER 0000


LA
LOAD ADDRESS

SR
SWITCH REGISTER 1414


DE
DEPOSIT

SR
SWITCH REGISTER 0000


LA
LOAD ADDRESS

SS
SINGLE STEP 1
```

pour une exécution cycle après cycle.

```
ST
START

PC
0001

MA
0000

MB
1414

L
0

AC
0000

IR
I

ET
FETCH

CO
CONTINUE

PC
0001

MA
0014

MB
```

visualisation des registres

```
AC
0000

IR
1

ET
DEFER

CO
CONTINUE

PC
0001

MA
4224

MB
1571

L
0

AC
1571

IR
1

ET
EXECUTE

PU
```

← visualisation du pupitre.

```
            PC 0001     TAD    EXECUTE
            MA 4224
            MB 1571
      L 0   AC 1571
            SR 0000      SS 1    SI 0


SI
SINGLE INSTRUCTION 1


SR
SWITCH REGISTER 0000


LA
LOAD ADDRESS

EX
EXAMINE 0000
1414

EX
EXAMINE 0001
6704

EX
EXAMINE 0002
3234
```

← examen mémoire.
en 0001, il y a 6704

```
S***  ←——————————————————— appel du superviseur.
      SEXECU  MONIT  ← appel de la phase.

MONITEUR

      .ED     (appel de l'éditeur).

EDITEUR


      *10         (insertion après la ligne - fictive - 0 : création d'un fichier).
BEFF   *
BEFF   * 00
       * 0100
       DEE    7200
              5000   TUT
       NB1    0342
       TUT    1000   DEE
              1000   NB1
              5400   AD1
       AD1    0200
       * 0200
              5600   AD2
BEFF   AD2    04100
       AD2    0410
       * 0410
              1200   NB2
              1200   NB3
BEFF          33
              3200   RESU
       NB2    4211
       NB3    5201
       RESU   0000
       F  ←————————————— fin d'insertion.


      *L1,20     ( demande de la liste des lignes 1 à 20)

       * 0100
       DEE    7200
              5000   TUT
       NB1    0342
       TUT    1000   DEE
              1000   NB1
              5400   AD1
       AD1    0200
       * 0200
              5600   AD2
       AD2    0410
       * 0410
              1200   NB2
              1200   NB3
              3200   RESU
       NB2    4211


      *L20,22

       NB2    4211
       NB3    5201
       RESU   0000


BEFF   *I1
        117         } insertion après la ligne 17
        FIN    7402
```

EXEMPLE 3

phase MONIT

```
*L1,23
        * 0100 <-------- pseudo-instruction pour le compteur d'assemblage.
DFE     7200            . 0000 → AC
        5000    TUT     . saut en TUT
NB1     0342            . valeur
TUT     1000    DEE     . [AC]+[DEE] → AC
        1000    NB1     . [AC]+[NB1] → AC
        5400    AD1     . saut en [AD1]
AD1     0200            . valeur
        * 0200 <-------- pseudo-instruction
        5600    AD2     . saut en [AD2]
AD2     0410            . valeur
        * 0410 <-------- pseudo-instruction
        1200    NB2     . [AC]+[NB2] → AC
        1200    NB3     . [AC]+[NB3] → AC
        3200    RESU    . [AC] → RESU
FIN     7402            . instruction d'arrêt
NB2     4211            . valeur
NB3     5201            . valeur
RESU    0000            . valeur

*F <-------- fin d'édition ; retour au moniteur.

FIN EDITEUR
MONITEUR

        .AS <-------- appel de l'assembleur.

ASSEMBLEUR-CHARGEUR
        OPT- L <-------- L, pour demander l'impression de la liste d'assemblage.

        0100   7200  DEF    7200
        0101   5103         5000   TUT
        0102   0342  NB1    0342
        0103   1100  TUT    1000   DEE
        0104   1102         1000   NB1
        0105   5506         5400   AD1
        0106   0200  AD1    0200
        0200   5601         5600   AD2
        0201   0410  AD2    0410
        0410   1214         1200   NB2
        0411   1215         1200   NB3
        0412   3216         3200   RESU
        0413   7402  FIN    7402
        0414   4211  NB2    4211
        0415   5201  NB3    5201
        0416   0000  RESU   0000
        FIN D'ASSEMBLEUR
MONITEUR

        .EX <-------- appel du simulateur

EXECUTION-P.D.P. DISPONIBLE
            MEMOIRE DISPONIBLE DE 0000 A 0577


        SI
        SINGLE INSTRUCTION 1        ) mode d'exécution : instruction par instruction


        SR
        SWITCH REGISTER 0100        ) 0100 : adresse de départ

        LA                          ) charge dans le PC
```

liste d'assemblage.

```
START    <------------------------  exécution 1ère instruction
AC
1000
CO
CONTINUE <------------------------  exécution 2ème instruction
FT
FETCH
MB
5103
CO
CONTINUE <------------------------  exécution 3ème instruction
AC
7200
CO
CONTINUE <------------------------  exécution 4ème instruction.
AC
7542
CO
CONTINUE <------------------------  exécution 5ème instruction.
FT
DEFER
MB
0200
CO
CONTINUE <------------------------  exécution 6ème instruction.
FT
DEFER
MB
0410
CO
CONTINUE <------------------------  exécution 7ème instruction.
BFFF     AA
         AC
         3753
CO
CONTINUE <------------------------  exécution 8ème instruction.
AC
1154
CO
CONTINUE <------------------------  exécution 9ème instruction.
AC
0000
IR
3
PU
```

```
          PC 0413    DCA    EXECUTE
          MA 0416
          MB 1154
L 0       AC 0000
          SR 0100        SS 0    SI 1
```

```
SR
SWITCH REGISTER 0416

LA
LOAD ADDRESS

EX
EXAMINE 0416
1154
```

} examen mémoire (résultat en RESU = 0416)

```
SR
SWITCH REGISTER 0100

LA
LOAD ADDRESS

SI
SINGLE INSTRUCTION 0

ST
START
```

} nouvelle exécution en mode "direct"

```
SR
SWITCH REGISTER 0416

LA
LOAD ADDRESS

EX
EXAMINE 0416
1154

F
```

← fin d'utilisation du simulateur. retour au moniteur.

```
FIN EXEC
MONITEUR

***
```

← simulation du fichier.

```
FIN MONITEUR
```

III § 5 - BILAN -

1°) - DEUX ANNEES D'EXPERIMENTATIONS -

(dans le certificat C2 de la maîtrise d'informatique de l'I.M.A.G.).

Deux années d'expérimentations mais trois années d'enseignement.

. La première année (1968/69).

C'était un enseignement tout à fait traditionnel.

" Programmation élémentaire en langage d'assemblage conversationnel sur P.D.P.8 en 12 séances hebdomadaires de 2 heures " disait le texte officiel.

Le " langage d'assemblage " est le langage de l'assembleur-correcteur CURAS du système SPARTACUS [C6]. Cet assembleur-correcteur est tout à fait anti-pédagogique [cf C7].

. Bilan de ce cours -

- La connaissance du fonctionnement du P.D.P.8 (et à plus forte raison du fonctionnement global d'un ordinateur quelconque) était quasiment nulle.

- La connaissance du fonctionnement de l'assembleur (et à plus forte raison d'un assembleur quelconque) était quasiment nulle.

- La connaissance du langage d'assemblage utilisé était imparfaite et de toute façon bien inutile.

- La connaissance des algorithmes traités était moyenne mais ils étaient très peu nombreux et très élémentaires (cela ne justifiait pas 24 heures de cours).

- Seule, la connaissance du mode d'emploi de l'assembleur-correcteur était acquise valablement au bout de ces 24 heures, mais c'était absolument inutile.

. La deuxième année (1969/70) -

Le projet OEDIPE avait vu le jour et le système SCALP commençait à se constituer.

- Mais l'enseignement était encore mal défini (Enseigner un peu la logique du fonctionnement du P.D.P.8. Enseigner un peu la programmation d'algorithmes en langage d'assemblage).
- Les moyens :
  - des premières versions des phases EXER et PROCES.
  - une première ébauche (timide) d'un manuel (→ M3).
  - une espérance d'assembleur P.D.P.8 sur le 360 d'I B M.
- Le temps disponible : 11 séances hebdomadaires de 2 heures.

Le cours :

. 7 semaines sur le fonctionnement de l'ordinateur (avec EXER et PROCES, versions 69).

. 1 semaine de cours sur les entrées-sorties (malheureusement, sans pratique).

. 3 semaines de programmation :
travail proposé : écriture d'un éditeur et d'un assembleur P.D.P.8.
L'assembleur espéré n'était pas opérationnel, et cette programmation est restée, malheureusement, sur le papier.

. Bilan de ce cours -

- La connaissance du fonctionnement du P.D.P.8 était moyenne.
- La connaissance du fonctionnement d'un software était bonne.
- La connaissance des algorithmes traités était encore imparfaite (mais les algorithmes traités étaient plus intéressants et plus complexes).

. La troisième année (1970/71) -

Le projet OEDIPE avait muri, et le système SCALP était, sinon terminé, tout au moins globalement défini.

- Le temps imparti était réduit à 20 heures.
- Nous avons proposé et obtenu l'emploi du temps suivant :
  - 5 semaines de cours.
  - 2 séances de 2 heures par semaine.
    . l'une, " sur machine ", était une séance de travail pratique. (1 maître, 1 groupe d'une quinzaine d'élèves).
    . l'autre, " hors machine ", était une séance de travail dirigé (2 maîtres, 1 groupe d'une trentaine d'élèves).
- Nous avons tenu, par principe, et en fonction de la réduction de temps, à nous borner à l'enseignement du fonctionnement logique de l'ordinateur et de son utilisation élémentaire.
- Les moyens :
  - Les phases EXER et PROCES (définitives).
  - L'espérance de la phase MONIT.
  - Un manuel de " synthèse " (les 3/4 de l'actuel M3).
  - Un ensemble audio-visuel (très artisanal, composé de diapositives, et commentées, en séance, par le maître).
  - Les contenus approximatifs (sinon les manuels) des manuels M1 et M2.

- Le cours :
  . Les 18 premières heures se sont passées sans problème et le programme fixé a été rempli (ce programme correspondait, à peu de choses près, à l'analyse diachronique décrite).
  . Pour la dernière séance, hélas, la phase MONIT n'était pas terminée, et les deux dernières heures ont été perturbées. (La description d'un software était faite, mais il n'y a pas eu le travail pratique correspondant).

Travail pratique réalisé :

TP1 : phase EXER 1.
TP2 : phase EXER 2.
TP3 : programme de traitement de table avec la phase PROCES.
TP4 : programmes " assembleur-octal " et " dump " sur le P.D.P.8 (avec utilisation des transferts télétype ↔ ordinateur).
TP5 : programmes libres pour l'étude de l'assemblage par programme assembleur.

. Bilan de ce cours -

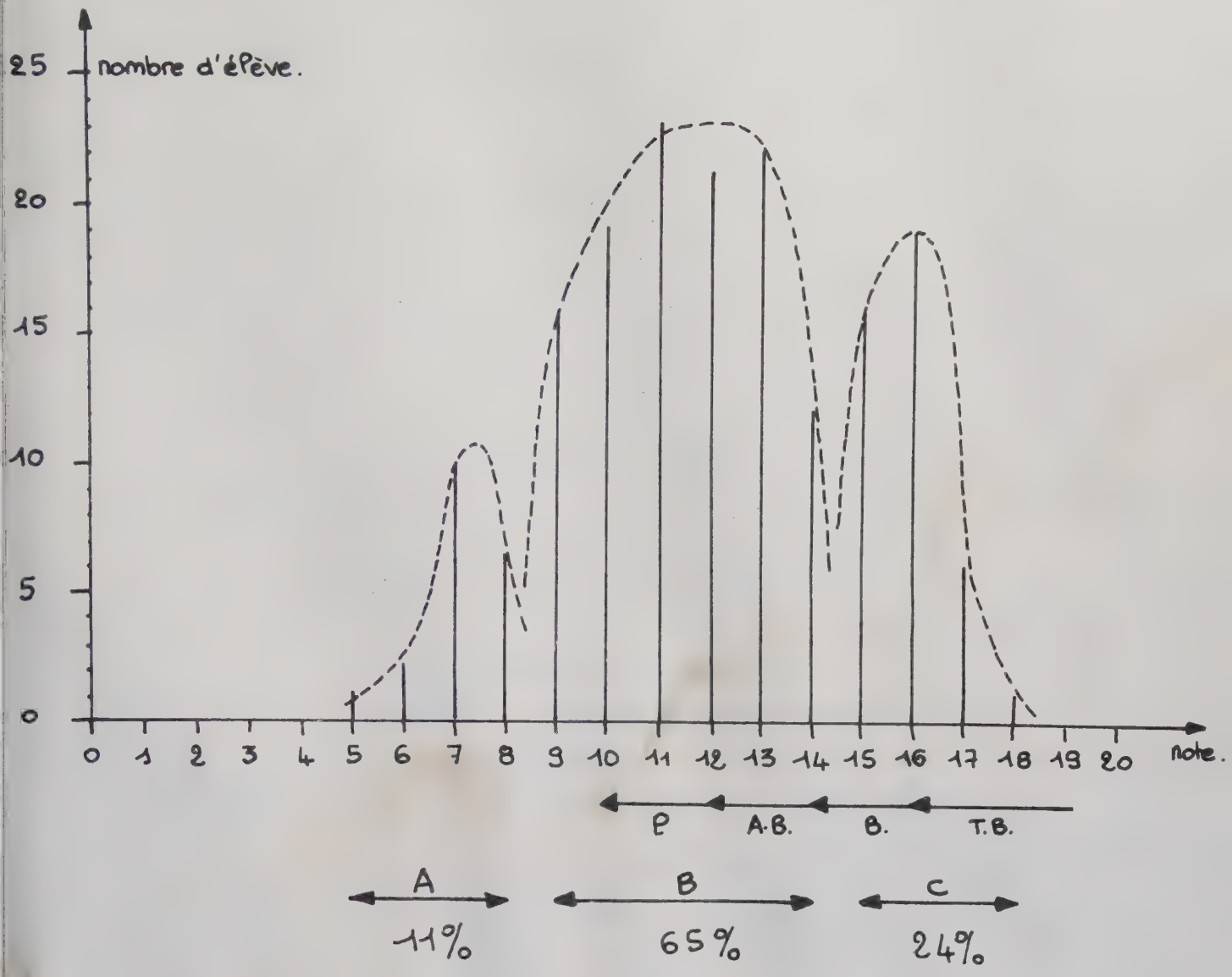
- Prévoir un programme cohérent pour 24 h et le réaliser en moins de 20 h, nous a conduit à faire un enseignement un peu rapide.
- L'absence de la phase MONIT était tout à fait regrettable.
- Connaître les contenus de M1 et de M2, ce n'était pas suffisant. Avoir les manuels M1 et M2 aurait été plus profitable. De même le dernier quart de M3 n'aurait pas été inutile.
- L'immobilité des diapositives nous a fait regretter, plus que jamais, le magnétoscope. (Corollairement, nous avons du en restreindre l'usage).
- Enfin, la phase NUMER, encore absente, aurait permis une meilleure pratique immédiate de la numération binaire/octale.

N.B. : un certain nombre d'élèves (faible heureusement) ne s'habitue que lentement au système " télétype-phase ". Et cette gêne peut réduire l'efficacité de la première séance " sur machine ". Si la première phase doit être inefficace dans certains cas, il vaut mieux que la première phase soit NUMER plutôt qu'EXER 1. (L'acquis est en effet moins important).

Cf Figure III 4.

FIGURE III. 4

RESULTATS



note < 10 : 19%
10 ≤ note < 12 : 24%
12 ≤ note < 14 : 24%
14 ≤ note < 16 : 17%
16 ≤ note : 16%

} population de 175 élèves.

N.B. Seule, la population présente aux 3 contrôles est représentée.

23 élèves (soit 13% de la population représentée) n'ont pas subi ces 3 contrôles dont 12 sans motif : cette population de 12 (7%) est la population non motivée et non assidue -

- La population A (11 %) est la population mal orientée et motivée uniquement par le " diplôme " final.

Le travail intense et précis contrôlé continûment accroît l'écart entre cette population A et le restant de la population.

- La population C (24 %) est la population non seulement motivée mais aussi très active. L'enseignement donné a réduit le travail personnel en dehors des " séances " de cours. Cette " réduction " a permis à cette population une " recherche " extérieure efficace.

Cette recherche extérieure accroît, là aussi, l'écart entre cette population C et le restant de la population.

- La population B (65 %) est la population " simplement " motivée qui ne s'intéresse qu'aux connaissances présentées par le cours.

Les résultats sont satisfaisants puisque les 84 % ($10 \leq$ note ≤ 14) de cette population B ont atteint, globalement, les objectifs fixés.

2°) - UNE EXPERIMENTATION DEFINITIVE -

Le système SCALP (P.D.P.8) est maintenant achevé. Une expérimentation complète est en cours.

Le texte " Programmation Fondamentale sur P.D.P.8.-Guide de l'enseignant " [A4] explicite tout le travail accompli par les élèves et présente un bilan précis.

REFERENCES BIBLIOGRAPHIQUES

A) - Le projet OEDIPE.

B) - La rénovation pédagogique.

C) - Documentation technique.

A) - LE PROJET OEDIPE.-

[A1] : J. P. PEYRIN : " Une application des méthodes de l'enseignement assisté à l'enseignement de la programmation élémentaire ".
Revue " Enseignement programmé " - Vol 11 - DUNOD - 1970.

[A2] : J. P. PEYRIN : " Un système d'enseignement de la programmation élémentaire ".
Congrès de l'AFCET - Vol 11 - 1970.

[A3] : J. P. PEYRIN : " Regard sur l'enseignement ".
IMAG. 1971.

[A4] : J. P. PEYRIN : " Programmation Fondamentale sur P.D.P.8 - Guide de l'enseignant ".
IMAG. 1971.

[A5] : J. P. PEYRIN : " Les phases programmées du système SCALP ".
IMAG. 1971.

J. P. PEYRIN : Les manuels du système SCALP :

[A6] : " Programmation Fondamentale sur P.D.P.8 - Manuel d'introduction ".
IMAG. 1971.

[A7] : " Programmation Fondamentale sur P.D.P.8 - Manuel de travail pratique ".
IMAG. 1971.

[A8] : " Programmation Fondamentale sur P.D.P.8-Manuel de références ".
IMAG. 1971.

B) - LA RENOVATION PEDAGOGIQUE :

[B1] : J. ALLOUCHE " Enseignement programmé et techniques audio-visuelles "
revue " Enseignement Programmé " - Vop 12 - DUNOD - 1970.

[B2] : A.C. ATKINSON " Models for memory "
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B3] : C. BELLISSANT " Conception et compilation d'un langage pour l'écriture de cours ".
Thèse de 3ème cycle. IMAG 1970.

[B4] : H. BESTOUGEFF " Etude du dialogue homme-machine dans un environnement pédagogique ".
Thèse d'état - Faculté des Sciences de PARIS. 1970.

[B5] : B. CAUSSE " L'enseignement par calculateur d'un langage de programmation ".
Revue " Enseignement Programmé " - Vol 1 - DUNOD 1968.

[B6] : S. CECCATO "The operational awareness as a premise to programmed instruction ".
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B7] : O. CLOUZOT " La méthode d'analyse sémantique d'un contenu ".
Revue " Enseignement Programmé " - Vol 6 - DUNOD 1969.

[B8] : D. COULON " Conception et réalisation d'un programme d'enseignement assisté ".
Thèse de 3e cycle. Faculté des Sciences de PARIS. 1970.

[B9] : R. COUSINET : " L'enseignement utilitaire ".
Revue " éducation et développement " n° 42 1968.

[B10] : M.J. DELIGNE : " Langages d'écriture des cours en enseignement programmé ".
Rapport de DEA. CNET 1969.

[B11] L. D'HAINAUT : " Une tentative de perfectionnement des cours programmés à réponse construite par adjonction d'une mémoire et rédaction à l'aide de séquences fonctionnelles ".
Revue " Enseignement programmé " - Vol 3 - DUNOD 1968.

[B12] : J. DONIO : " Les éléments du contrôle en enseignement programmé ".
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B13] : W. FEURZEIG - S.A. PAPERT - " Programming Languages as a conceptual framework for teaching mathematics ".
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B14] : G.E. FORSYTHE : " Computer science and éducation ".
Proceedings of the IFIP Congress 68 - Vol 2 - 1968.

[B15] : C. FREINET - M. BERTELOOT - " Travail individualisé et programmation "
Bibliothèque de l'école moderne. 1966.

[B16] : B.R. GAINES : "Adaptively controlled instruction for a tracking skill "
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B17] : R.M. GAGNE : " Learning categories and instructional strategies "
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B18] : R. GLASER : " Categories of learning and criteria for evaluation of learning outcomes ".
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B19] : J. GRITTI : "Télévision, nouveau langage ".
Revue " MEDIA " - n° 9 - IPN 1970.

[B20] : K.E. IVERSON : " a common language for hardware, software and applications ".
Proceedings F. J. C. C. 1962.

[B21] : K.E. IVERSON : " The use of APL in teachning ".
I B M 320-0996 1969.

[B22] : D.E. KNUTH : " Fundamental Algorithms - The art of the computer programming ".
ADDISON-WESLEY 1968.

[B23] : L.N. LANDA : " The construction of algorithmic and heuristic models of thinking activity and some problems in programmed learning "
Aspects of Educational Technology - Vol 2 -
University of Glasgow. 1968.

[B24] : J. LEIF : " Les inspirations et tendances nouvelles de l'éducation ".
DELAGRAVE 1967.

[B25] : G.A. MARTIN : "Analyse logique des réponses en enseignement automatisé par calculateur ".
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B26] : J. MASSON " ORDI 2 "
Faculté des Sciences de Paris. 1969.

[B27] : J. MOURGEON : " Intégrer les circuits de télévision ".
Revue " MEDIA " n° 10. IPN 1970.

[B28] : P. OLERON : " Enseignement programmé, transmission et acquisition des connaissances ".
Revue " Enseignement Programmé " Vol 9-10 DUNOD 1970.

[B29] : G. PASK : " Adaptative machines ".
Tendances actuelles de la recherche en enseignement programmé.
Sciences du comportement. DUNOD 1968.

[B30] : J. PERRIAULT : " Inventaire des méthodes d'analyse des matières en vue de leur enseignement ".
Cours. UER. Didactique des disciplines scientifiques. PARIS. 1970.

[B31] : J. PIAGET : " Psychologie et Pédagogie ".
DENOEL 1969.

[B32] : L. PINSON - H. CHAMPETIER DE RIBES - J. LAISNE : " Un système d'enseignement programmé audio-visuel ".
Revue " Enseignement Programmé " - Vol 5 - DUNOD 1969.

[B33] : L. POHL : " Signification et importance des algorithmes dans l'enseignement des langues vivantes ".
Revue " Enseignement Programmé " - Vol 2 - DUNOD 1968.

[B34] : L. PORCHER " Education nouvelle et moyens audio-visuels ".
Revue " Education et Développement " n° 46 1969.

[B35] : A. PROFIT : " Structure et technologie des ordinateurs ".
Collection U. Armand COLLIN. 1970.

[B36] : C. ROGERS : " Le développement de la personne ".
(Articles parus entre 1951 et 1961) DUNOD 1968.

[B37] : A.W. SCHESTAKOW : " L'enseignement programmé et les machines à enseigner en U R S S ".
Sciences du Comportement. Vop 5 - DUNOD 1968.

[B38] : B.F. SKINNER : " La révolution scientifique de l'enseignement ".
CH. DESSART - BRUXELLES - 1968.

[B39] : D. SUPPES : " Computer Assisted Instruction : an overview of operations and problems ".
IFIP CONGRESS 68. Vol 2 1968

[B40] : F.M. TONGE : " Design of a programming language and system for computer assisted learning ".
IFIP CONGRESS 68. Vol 2 1968

[B41] : J. WITTNER : " Les illusions pédagogiques ".
Revue " Education et Développement " n° 42 - 1968.

[B42] : K.L. ZINN : "Languages for programming conversational use of computers in instruction ".
IFIP CONGRESS 68. Vol 2 1968

C) - DOCUMENTATION TECHNIQUE.

[C1] : DIGITAL EQUIPMENT CORPORATION : " Introduction to Programming ".
P.D.P.8 - Handbook Series. 1970.

[C2] : DIGITAL EQUIPMENT CORPORATION : " Programming languages "
P.D.P. 8 - Handbook Series. 1970.

[C3] : DIGITAL EQUIPMENT CORPORATION : " P.D.P.8 - P.D.P.8/I - P.D.P.8/E "
Small Computer Handbook 1967/68/69/70.

[C4] : DIGITAL EQUIPMENT CORPORATION : " Disk Monitor System ".
Programmer's reference manuel - P.D.P.8. Family - 1968.

[C5] : C. BOKSENBAUM - S. GUIBOUD-RIBAUD. " Etude et réalisation d'un système pédagogique de programmation en ligne et en temps partagé ".
Projet de fin d'études I. P. G. 1967.

[C6] : S. GUIBOUD-RIBAUD - " Le système SPARTACUS ".
IMAG 1968.

[C7] : P. JONES - " Repérage de fichiers sur SPARTACUS ".
Rapport de D E A IMAG 1969.

[C8] : IAN-BERGER : " Assembleur P.D.P.8 sur IBM 360 "
Projet de fin d'études I. P. G. 1969.

[C9] : I B M " CP/CMS User's Guide "
Cambridge Scientific Center 320-2015

[C10] : K.E. IVERSON : " The use of APL in teaching "
IBM - 320-0996 1969.